APPLICATION OF GAME THEORY FOR ACTIVE CYBER DEFENSE AGAINST ADVANCED PERSISTENT THREATS

Benn, Andrew S.; Benn, Stephanie M.

Monterey, CA; Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

APPLICATION OF GAME THEORY FOR ACTIVE CYBER DEFENSE AGAINST ADVANCED PERSISTENT THREATS

by

Andrew S. Benn and Stephanie M. Benn

September 2023

Thesis Advisor: Neil C. Rowe
Co-Advisor: Wade L. Huntley

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2023	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
APPLICATION OF GAME THEORY FOR ACTIVE CYBER DEFENSE AGAINST ADVANCED PERSISTENT THREATS

5. FUNDING NUMBERS

6. AUTHOR(S) Andrew S. Benn and Stephanie M. Benn

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

Advanced persistent threats (APTs) are determined, adaptive, and stealthy threat actors in cyber space. They are often hosted in, or sponsored by, adversary nation-states. As such, they are challenging opponents for both the U.S. military and the cyber-defense industry. Current defenses against APTs are largely reactive. This thesis used machine learning and game theory to test simulations of proactive defenses against APTs. We first applied machine learning to two benchmark APT datasets to classify APT network traffic by attack phase. This data was then used in a game model with reinforcement learning to learn the best tactics for both the APT attacker and the defender. The game model included security and resource levels, necessary conditions on actions, results of actions, success probabilities, and realistic costs and benefits for actions. The game model was run thousands of times with semi-random choices with reinforcement learning through a program created by NPS Professor Neil Rowe. Results showed that our methods could model active cyber defense strategies for defenders against both historical and hypothetical APT campaigns. Our game model is an extensible planning tool to recommend actions for defenders for active cyber defense planning against APTs.

14. SUBJECT TERMS advanced persistent threats, game theory, active cyber defense, cyber deception, moving-target defense, reinforcement learning, machine learning	**15. NUMBER OF PAGES** 105
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

APPLICATION OF GAME THEORY FOR ACTIVE CYBER DEFENSE AGAINST ADVANCED PERSISTENT THREATS

Andrew S. Benn
Captain, United States Marine Corps
BS, United States Naval Academy, 2017

Stephanie M. Benn
Captain, United States Marine Corps
BS, United States Naval Academy, 2017

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2023

Approved by: Neil C. Rowe
Advisor

Wade L. Huntley
Co-Advisor

Gurminder Singh
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Advanced persistent threats (APTs) are determined, adaptive, and stealthy threat actors in cyber space. They are often hosted in, or sponsored by, adversary nation-states. As such, they are challenging opponents for both the U.S. military and the cyber-defense industry. Current defenses against APTs are largely reactive. This thesis used machine learning and game theory to test simulations of proactive defenses against APTs. We first applied machine learning to two benchmark APT datasets to classify APT network traffic by attack phase. This data was then used in a game model with reinforcement learning to learn the best tactics for both the APT attacker and the defender. The game model included security and resource levels, necessary conditions on actions, results of actions, success probabilities, and realistic costs and benefits for actions. The game model was run thousands of times with semi-random choices with reinforcement learning through a program created by NPS Professor Neil Rowe. Results showed that our methods could model active cyber defense strategies for defenders against both historical and hypothetical APT campaigns. Our game model is an extensible planning tool to recommend actions for defenders for active cyber defense planning against APTs.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACM	attack-centric method
AD	active defense
AI	artificial intelligence
APT	advanced persistent threat
C2	command and control
CCAT	Cyberspace Course of Action Tool
CISA	Cybersecurity and Infrastructure Security Agency
CJCSM	Chairman of the Joint Chiefs Staff manual
CSP	cloud-service provider
CSV	comma-separated values
DCO-IDM	defensive cyber operations – internal defense measures
DNS	Domain Name Service
DoD	Department of Defense
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
ICS	industrial control system
IDS	intrusion-detection system
IIoT	industrial Internet of things
IoT	Internet of things
IPS	intrusion-prevention system
MAC	Media Access Control
ML	machine learning

NIST	National Institute of Standards and Technology
RDP	Remote Desktop Protocol
RSA	Rivest Shamir Adleman
RST	reset flag
SCADA	Supervisory Control and Data Acquisition
SCAMP	Social Causality using Agents with Multiple Perspectives
SMTP	Simple Mail Transfer Protocol
SQL	Structured Query Language
SSH	Secure Shell
SYN	synchronize flag
SYN-ACK	synchronize-and-acknowledge flag
TCP/IP	Transport Control Protocol/Internet Protocol
t-SNE	t-stochastic nearest-neighbor embedding
URL	Uniform Resource Locator

ACKNOWLEDGMENTS

Thank you, Professor Rowe, and Professor Huntley, for guiding us through this intellectual challenge. We appreciate the freedom you gave us to explore our interests, as well as the many course corrections to keep us on track. We could not have done this without your intentional support and direction.

Thank you to our parents for the moral support and encouragement these past few years. We endeavor always to make you proud.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Advanced persistent threats (APT) are critical issues for both the cyber-defense industry and the Department of Defense (DoD). These well-resourced, stealthy, and skilled actors adapt to defenses they encounter, and use many techniques to accomplish their objectives. A defender wishing to resist an APT must build a consistent, multilayered defensive system to detect and address incursions. Current approaches to cyber defense against APTs are reactive. They focus on detecting an attack action, and then countering with a defender reaction. These models cannot adequately represent proactive-defense ("active-defense") measures.

This thesis uses machine learning and game theory to produce a model that can support defender proactivity against APTs. This research supports U.S. National Cybersecurity Strategy principles of advancing defense of critical infrastructure by making federal systems more defensible and resilient (U.S. White House, 2023).

A. IMPORTANCE TO THE U.S. MILITARY

The U.S. Department of Defense (DoD) identifies cyberspace as a warfighting domain in which the United States must compete and distinguishes a secure cyberspace as a critical national interest (U.S. White House, 2022). Military information networks, information systems, and defense industrial-base networks operate in cyberspace, and therefore are vulnerable to many kinds of cyberattacks (CJCSM Cyber Incident Handling Program, 2018). United States military networks are probed or scanned 250,000 times per hour by more than 100 intelligence agencies and foreign militaries (Kahn et al., 2011), and subsequent attacks and intrusions cause loss of sensitive information such as military intelligence and plans through data exfiltration (Bell, 2019). Military networks are vulnerable because their information-technology infrastructure is usually tied to civilian infrastructure that "could be targeted directly in a conflict or be held hostage as a bargaining chip against the U.S. government" (Lynn, 2010).

Within military cyberspace, APTs are particularly dangerous. An APT is a cyber actor that uses sophisticated, targeted, and highly organized cyberattacks (Quintero-Bonilla

et al., 2020). Attacks by APTs can be difficult to detect and attribute. This is because large political entities and organizations (state, state-sponsored, and non-state) perpetrate these attacks, and because attackers often masquerade as a third party by mimicking its attack features with Internet addresses, email addresses, and malicious code. The National Institute of Standards and Technology (NIST) says an APT pursues "its objectives repeatedly over an extended period of time … [adapts] to defender's efforts to resist it … maintains the level of interaction needed to execute its objectives" (Joint Task Force Transformation Initiative, 2011). As APTs of geopolitical adversaries develop their capabilities for future conflicts, it is important that the United States develop active cyber defense measures to counter them. Table 1 shows examples of APTs with which the U.S. military is concerned.

Table 1. APTs that are known to target military entities. Sources: Mandiant, (2022a), Mandiant (2022b), and Pingios et al. (2022).

APT	Suspected Attribution	Satisfaction of APT Characteristics	Goals
APT14 (Mandiant, 2022a)	China	APT14 focused on targeting military and maritime equipment, operations, and policies by using custom Simple Mail Transfer Protocol tools to send spear-phishing messages that appeared to come from trusted organizations (Mandiant, 2022).	APT14 sought to improve Chinese military operations and interfere with rival military communication networks through data theft.
APT29 (Mandiant, 2022b)	Russia	APT29 demonstrated persistence by adding new service users with administrator permissions, staging files in password protected archives, creating custom backdoors, and running complex and extended spear-phishing campaigns. It avoided detection by layers of remote execution, used steganography to hide payloads, used custom malware, encrypting C2, and renamed malicious dynamic link libraries and executables with benign names (Mandiant, 2022).	APT29 sought to influence domestic politics (Democratic National Committee hack) and compromised supply chains (SolarWinds attack). They continue to seek to collect intelligence on military and government entities.
APT35 (Pingios et al., 2022)	Iran	APT35/Magic Hound demonstrated persistence through registry modification and administrator-account creation. They avoided detection by disabling antivirus software, editing firewall rules to allow Remote Desktop services, deleting and overwriting log files to cover their tracks, using complex social engineering campaigns, and clearing command histories after an attack (Pingios et al., 2022).	APT35 targeted military and defense industries by exporting emails and credentials, establishing remote control on machines, and compromising domain controllers. This supported Iranian national aims and intelligence collection.

B. BACKGROUND

APTs are asymmetric threats because they can use many attack vectors to access a network, including zero-day exploits, social engineering, spear phishing, and drive-by-downloads (Quintero-Bonilla et al., 2020). Once an APT accesses a network, it can use

customized malware to maintain network access, move throughout the system, and detect and exploit additional vulnerabilities (Mandiant, 2021).

1. Addressing APTs Systematically

Because APTs are varied and complex, defending against them has been largely reactive and based on detection of individual exploits. Some steps in mitigating the threat posed by APTs are:

- Improve security policy and controls providing communication and systems security (Joint Task Force, 2021). The goals of communication security are to provide confidentiality, data integrity, and peer authentication. The goals of systems security are to prevent unauthorized usage, inappropriate usage, and denial of service.

- Monitor current APT campaigns (Marchetti et al., 2016) to learn what methods they are using. This is done by pattern matching of attack signatures and methods. APTs may mimic normal behavior and carefully target hosts to remain undetected.

- Once an APT is identified, allocate resources to fix the compromised hosts following a mostly predefined response plan (Yang et al., 2018).

- Assess the APT threat (Mell et al., 2006). A standard assessment strategy is based on the MITRE Common Vulnerability Scoring System, which generates an overall vulnerability score with base, temporal, and environmental metric groups. These groups assess the effects of the threat based on confidentiality, integrity, availability, exploitability, and collateral-damage potential, among other factors.

- Include cyber deception in the response strategy. Techniques such as LaBrea (Haig, 2022), which delay rogue machines on a network, and hardened honeypots (Meier et al., 2023), which act as decoys to log

information about attack patterns, entry points, and malware types as attackers access them, are possible tactics for defending against an APT.

Knowing when systems are under attack is essential to a timely and effective response. Intrusion-detection systems (IDSs) and intrusion-prevention systems (IPSs) can do this by anomaly-based or signature-based monitoring to find attacks on network and host devices (Bazrafshan, 2013). They can use artificial-intelligence techniques to determine if a disguised packet is malicious and can learn of new threats with machine learning. Signature-based monitoring identifies malware or malicious activity based on a database of hashes of suspicious patterns of bytes; hashes must be constantly updated to keep pace with new threats and attacks. Anomaly-based monitoring compares network traffic with an established baseline and identifies traffic that deviates from it.

2. Improving Detection of APTs

Reactive approaches to defending against APTs are slow and require time to fix affected systems, which impacts operations and has monetary costs. Because new attack methods are created all the time, machine learning can accelerate APT detection-clue formulation. Machine learning is a subfield of artificial intelligence that automatically builds new models of classes of data (National Science Technology Council, 2020) and can support automated learning of clues for identification of patterns of attack.

Furthermore, real-time machine learning could enable more options for active defense systems against APTs. Active defenses can also be developed using game theory, which models interactions and decisions among two parties with conflicting goals. Game theory provides a framework of players, actions, payoffs, and strategies (Ferguson-Walter et al., 2019). Examining attacker and defender interaction with a game theory model enables defenders to develop and test deception, delaying tactics, and other active-defense measures to lengthen the interaction and prevent attackers from achieving their goals (Liang & Xiao, 2013). These mathematical models simulate real-world conditions and emotions, and include uncertainty, imperfect knowledge, and regret. The models can produce useful tools for planning and understanding these factors.

Both machine learning and game theory have been studied for defensive deception in cyber defense (Zhu et al., 2021). However, limited work has been done with these approaches for APTs, whose attacks are complicated and multi-step. Also, limited work has been done combining both machine learning and game theory in hybrid defensive deception. This research develops a hybrid approach of combining them using reinforcement learning with game theory. This is done by applying ensemble machine learning to two benchmark APT datasets to recognize attack steps in traffic by attack stage. The derived observations from classification are then included in a game-theoretic active defense for an active cyber defense game using realistic costs.

3. Thesis Organization

The rest of this thesis has these chapters:

- Chapter II, Literature Review, describes previous attempts at APT modeling and other problems like it.
- Chapter III, General Approach, describes the setting of the problems we address, the datasets used in our detection research, and a use case for a hypothetical APT to build a defender specification.
- Chapter IV, Game Modeling Methodology, describes the processes we used to design inputs to the game-modeling program, and describes the game-modeling program that uses reinforcement learning.
- Chapter V, Results and Discussion, summarizes performance of our game model, and discusses insights and quantifiable results.
- Chapter VI, Conclusion, states major achievements and limitations of our research, assesses the usefulness of our game model, and makes recommendations for future work.

II. LITERATURE REVIEW

A. APT MODELING

Threat modeling generates realistic representations of cybersecurity threats to assess system vulnerabilities and potential effects (Tatam et al., 2021). APT threat modeling must include features specific to APTs such as their high levels of resourcefulness, adaptability, and persistence. This requires understanding attacker tactics, techniques, and procedures (Nweke & Wolthusen, 2020).

APT threat modeling can be asset-centric, system-centric, threat-centric, or data-centric (Tatam et al., 2021). Asset-centric approaches are based on risk analysis and asset criticality. System-centric approaches are based on identifying components of a system's design and its software that are vulnerable to attack. Threat-centric approaches focus on identifying attack vectors and targets. Data-centric approaches focus on protecting data within systems (Murugiah & Scarfone, 2016).

APTs were originally modeled using the phase-based cyber kill chain (Hutchins et al., 2011). This is the systematic process by which an APT typically targets and engages their adversaries. Any disruption of a phase of an APT attack should disrupt the entire attack. Unfortunately, this kill-chain model cannot represent more sophisticated APTs in which actions are not so rigid and patterned. An alternative is the Diamond model which focuses on the fundamental elements of malicious activity: an adversary using some capability, through infrastructure, to attack a victim (Caltagirone et al., 2013).

Because APTs must penetrate many layers and exploit many vulnerabilities of a system, they can be modeled as a multi-stage and multi-phase game (Zhu & Rass, 2018). It used Bayesian methods to model spear phishing, sequential nested games to model multi-stage penetration of networks, and a finite zero-sum game to model physical-layer infrastructure protection. However, it did not consider specific active defensive techniques. This model focused on preventing loss, rather than gaining information and time for the defender. The work did not model the uncertainty needed to learn the game's optimal (Nash) equilibrium.

One study modeled the APT life cycle with a multi-level attack-defense tree (Fei et al., 2018). Another study the APT as an attacker pyramid, where lateral planes represent attack environments such as the physical, user, network, and application ones (Giura & Wang, 2012). These studies primarily focused on detection as defensive measures.

Other work applied a two-layer model (domain and kill-chain) to the Stuxnet APT attack to find system chokepoints as good places for deploying countermeasures (Kumar et al., 2021). This counters an APT's attacks with a layered defense-in-depth strategy. As vulnerability assessment alone cannot prevent APTs from successfully attacking, chokepoint analysis should include defenses at all APT stages.

Other efforts used hidden Markov models to represent an APT as a continuous campaign (Brogi & Di Bernardino, 2017). This helped reconstruct campaigns undetected by an intrusion-detection system, which is useful because detection of an APT is difficult.

A weakness of all APT modeling is that the model should be validated with a real-life dataset, something currently difficult because of the limited public real-life datasets for research.

B. APT DETECTION

Initial stages of an APT's life cycle involve reconnaissance of the target network. This generates network traffic that can be detected. Furthermore, as remote control of APTs is essential, the necessary external communications create more traffic for detection. One study found that 90% of APT control was done with the Web protocol HTTP (Wang et al., 2016). Some clues permit distinguishing these communications from normal HTTP requests. These clues were applied to Domain Name Service records and validated on a dataset provided by the Los Alamos National Laboratory, though detection could be evaded by compromised hosts.

Other studies focused on detecting APTs within specific stages of their lifecycles. APTs using the Remote Desktop Protocol (RDP) for lateral movement on a local-area network can be detected (Bai et al., 2019), and other clues enable detection at their subsequent stages (Aparicio-Navarro et al., 2018). However, these studies require

observations of a network's normal operating baseline, which differs considerably between systems. Another study focused on long-term APT campaigns, which are often customized for the targets and take months to perform (Brogi & Tong, 2016). The TerminAPT or APT detector tool used information-flow tracking to link the stages of an attack campaign and traces left by attackers.

Some research focused on Transport Control Protocol/Internet Protocol (TCP/IP) connections in APTs, examining the number of packets transferred, the duration of connections, and other packet-flow details (Siddiqui et al., 2016). Performance was improved with a Hadoop distributed-processing architecture (Shenwen et al., 2016). Another study compared detecting and classification techniques of APTs using support-vector machines on the Knowledge Discovery and Data mining dataset, which has examples of both benign traffic and APT reconnaissance (Chu et al., 2019).

C. MODELING DEFENSE AND DECEPTION

1. Considerations for Deception in Modeling Cyber Defense

Many deception techniques can interfere with or defeat adversaries during the phases of an APT (Virvilis et al., 2014). Some techniques proposed from one study include darknets (computers that receive communication but do not respond), honeynets (decoy targets, sometimes with intentional vulnerabilities), database honeytokens (fake records that make it easier to track attacks), honeyfiles (files designed to detect access), and honey accounts (accounts that collect details of interaction with attackers for analysis).

Other studies modeled effectiveness of deception methods in planning "counter-deception" for defending information systems (Rowe & Rrushi, 2016, p. 161-171). This trust model used generic excuses for deception and a Naïve Bayes approach to calculate likelihoods of different hypotheses about the attacker. It also estimated the best generic excuses for deception, and constructed counterplans to discourage an adversary. An alternative approach used two formulas, a legitimate-user penalty, and an excuse delay (Rowe & Rrushi, 2016, p. 172-180). It was used to generate a Markov state graph from a rootkit attack model and provide a basis for deception. Generic excuses can convince an attacker that their attacks are not working.

Another project studied delays as a deception tactic (Rowe, 2007a). A defender can also falsely claim temporary unavailability of critical resources. This can overrun the attacker's patience and waste the time allocated for the attack. Another project used multilayer cyber deception to detect sophisticated attacks (Wang et al., 2013). This study used deception-based detection to increase the probability of attacker detection. Their model distinguishes "honey people", "honey files with honey activity", and "honey servers with honey activity".

2. Approaches to Modeling Cyber Defense and Deception

Researchers have modeled cyber defense in several ways. One method used two simultaneous game trees, one for the attacker and one for the defender (Ferguson-Walter et al., 2019). When the attacker used a "greedy" (or opportunistic) strategy in experiments, their payout was not as high as expected due to the defender. This study recommended that research should investigate attack trees tied to defender goals as opposed to attacker actions.

Hypergames model the players who can have misperceptions of a conflict (Xi & Kamhoua, 2020). This work assumed that the defender used only a honeypot, to slow attacks and gather adversary information. For this model, a mixed equilibrium strategy (varying tactics randomly) was best for both players. Analysis of best honeypot placement was also done.

Another study modeled interactions between an attacker and defender using minimax methods (Lin et al., 2009). The researchers examined a zero-sum game modeling a common network environment with assigned weights from the attacker's viewpoint, a payoff table of strategies, and a probability distribution that the results satisfied both players. One more study used finite-state machines to model three types of cyber defensive deception: basic deception processes, deception-incorporation models, and cyber denial and deception models (Hassan & Guha, 2016). The authors noted limited effectiveness of their techniques against well-resourced attackers such as APTs in which the deception would never reach its final state.

3. Modeling Defense against APTs

A study examined generalized matrix games with uncertainty and unknown incentives specifically for APTs (Rass et al., 2017). With a distribution for payoff calculation, they included uncertainty in the model by making the payoff of an action unknowable until it was chosen. They also used continuous rather than sequential moves for both attacker and defender to model the mutual uncertainty about the other's incentives.

Other research showed that honeypots effectively defend against APT attacks on cyber-physical systems with limited resources (Tian et al., 2019). This was done by proving optimal defensive strategy for low-interaction and high-interaction honeypot modes. Resource-limitation considerations included honeypot allocation and human analysis costs. Another study used a Stackelberg (sequenced leader-follower) game model, which can better describe an APT's attack life cycle than that of a Nash game in which players act simultaneously (Bi et al., 2022). The researchers examined an industrial Internet of Things system during the APT's lateral-movement stage. Results were compared with both greedy and random algorithms and showed that the model was an improvement. Another study proposed a security testbed for modeling attacks against industrial control systems for the energy industry (Park et al., 2021). The researchers used a real-time simulation of industrial control systems and penetration-testing tools with APT-style attacks against the system. Yet another study designed a game in which attacker and defender competed to control resources within an industrial control system (Rubio et al., 2020). The work showed that opinion dynamics, a conceptual model agent influences over others, can effectively reduce an APT's impact against critical infrastructure.

4. Modeling Tools

Tools for building security games provide implementations of mathematical models with computer software. Some have been adapted from models to simulate social conflicts, and others are specifically designed for cyber threats.

One tool, Social Causality with Agents using Multiple Perspectives (SCAMP), is a social simulator in which agents make choices (Parunak et al., 2021). It produces a directed graph of events that describe an agent's game experience. Features of the event nodes

identify tactical decisions made by the agent. Hierarchical goal networks evolve as the game handles new goals.

A proprietary tool for modeling military decision-making in the cyber domain is Cyberspace Course of Action Tool (CCAT) (Green et al., 2020; Drew et al., 2021). CCAT was developed by Soar Technology Incorporated and simulates and tests the effect of deception and active methods within a computer network, modeling both attacker and defender with game theory. A similar tool is the MITRE tool CALDERA which simulates autonomous adversary emulation, autonomous incident response, and manual red-team engagements (MITRE, 2023a). It includes agents, reporting, and collection of tactics to automate security assessments to save time, money, and energy.

Another specific deception planner was a program that planned convincing false excuses about availability of system resources (Rowe, 2007b). It computes the best deceptions based on the commands executed in an operating system.

The Cybersecurity and Infrastructure Security Agency offers a tool Decider that maps threat actor behavior to the MITRE ATT&CK framework (CISA, 2023). This Web application provides a framework to which defenders can add attack tactics from the ATT&CK knowledge base, to help defenders better understand attacker behavior.

D. SUMMARY

Many APT models are rigidly sequenced or tightly patterned, and do not capture the flexibility that APTs display in countering defenses. They often treat APTs identically and generically. Models often do not consider defensive techniques that can be applied against an APT, as their primary focus is on detection, not active defense. Detection largely focuses on determining whether traffic is malicious or not, rather than attempting to classify traffic to determine appropriate responses. When approaches use specific tactics, they usually focus on one phase of an APT. Defensive techniques generally focus on one defensive action class (e.g. deception, detection, or moving-target defense). Often, they are narrowed to specific systems (e.g. energy systems, or industrial control systems) rather than being general models.

We address these gaps by proposing a game model in which attackers and defenders engage repeatedly while using a variety of techniques across many action classes and attack phases. Our defenders and attackers also adjust their tactics based on feedback through reinforcement learning.

THIS PAGE INTENTIONALLY LEFT BLANK

III. GENERAL APPROACH

A. DETECTING THREATS

Before modeling defense and deception against APTs, we must understand what an APT attack looks like to the defender: a network traffic stream that contains the attack within legitimate operations, mixed with administrative traffic required for the target and its network to function. The mixture of malicious and legitimate traffic complicates detection, making APTs a good candidate for machine learning to find patterns that humans have difficulty recognizing.

APT attack life cycles typically contain these phases in loose sequence:

- Phase 1, do reconnaissance: Attackers gather credentials and other pieces of information about their target to support subsequent phases.
- Phase 2, establish foothold: Attackers successfully enter the target's network.
- Phase 3, move laterally: Attackers search for critical components and sensitive organizational data. They try to ensure long-term (persistent) access to the target network.
- Phase 4, exfiltrate data: Attackers get data they want, usually by moving it from the target system to their command-and-control center.
- Phase 5, pivoting: Attackers may use the network compromise to stage attacks on another network.

Two datasets for APTs generated in 2020 and 2021, DAPT2020 and SCVIC-APT-2021, include all stages of the APT attack life cycle (Myneni et al., 2020, and Liu et al., 2022). These datasets are also especially useful because they contain real network traffic rather than traffic created by a traffic generator (synthetic) or a mixture of the two (hybrid). Table 2 compares these datasets with other public datasets for APTs.

Table 2. Public APT attack datasets by attack phase.

Dataset Name	Capture Type	Benign Traffic	Recon-naissance Phase	Establishing Foothold/ Initial Compromise Phase	Lateral Movement Phase	Data Exfiltrat-ion Phase	Pivoting Phase
NSL-KDD	Hybrid	X	X	X			
UNB-15	Syntheti c	X	X				
CICIDS	Syntheti c	X	X	X			
DAPT 2020	Real	X	X	X	X	X	
SCVIC-APT-2021	Real	X	X	X	X	X	X

The datasets try to classify traffic by attack stage. For DAPT2020, this was the result of semi-supervised machine learning (Myneni et al., 2020). Classifying APT traffic is difficult, especially classifying an APT as a whole, rather than its phases. Machine-learning techniques used for DAPT2020 included support-vector machines, stacked autoencoders, and long-short-term-memory neural networks. For the SCVIC-APT-2021 dataset, the creators applied an "attack-centric" ensemble (multi-technique) method (Liu et al., 2022) to identify attack phases, selecting features best for identifying each type of attack. Several rounds of binary classifiers were then trained, one for each type of attack. This technique was combined with random undersampling of legitimate traffic to produce its model for classification. A random-forest classifier was also tried for a 2% improvement.

Recognizing attack stages is also important for the defender to enable them to anticipate what an attacker is likely to target. This anticipation includes attacker and defender techniques that are likely to be used, and an automated defense can quickly confirm which phase of an attack is ongoing so it can deploy the correct countermeasures.

Similarly to the DAPT2020 and SCVIC-APT-2021 researchers, we sought to detect APT activity by life-cycle stage. A defender can be more effective if they determine the

best stage to deploy effective defenses. We classified by combining voting classifiers and near-miss undersampling.

1. More Details of the Datasets

The DAPT2020 dataset simulated APT behavior by executing Metasploit and other attack tools on a sample network to simulate attack, compromise, and exploitation by an APT (Kennedy et al., 2011). Attacks were done alongside benign traffic on a lab network, collected using the Tcpdump and Snort intrusion-detection tools to capture the live traffic over a work week. The data is in CSV files of 72,334 rows with 83 features and two labels (activity and stage). It contains traffic features from packet captures, Syslog logs, DNS logs, Apache logs, authentication logs, database-server logs, host intrusion-detection logs, and network intrusion-detection logs. The columns are all numeric, including code numbers of communication protocols, flow durations, packets counts, packet sizes, traffic rates, average times between packets, and flag counts, etc.

SCVIC-APT-2021 expanded on the methods of the DAPT2020 dataset to include additional stages like pivoting (Liu et al., 2022). Several additional attack types were implemented and captured using CiCiFlowMeter. The data also includes Internet-of-Things devices.

2. Phase Classification Observations

Using Pandas, Seaborn, Yellowbricks, and Sci-Kit Learn libraries for the Python programming language, we cleaned the data. Unique information for each record such as the IP socket pair and timestamps were removed. Stage names were numbered. Using variance as a filter, all features with a variance less than 0.1 were dropped. This was done to eliminate several features in the dataset that had a constant value of zero for all entries or were unused by the dataset (e.g. Fwd Bulk Rate Avg, Bwd Bulk Rate Avg, Bwd Bytes/ Bulk Avg, Bwd Packet/Bulk Avg, Fwd Seg Size Min, Fwd Init Win Bytes, URG Flag Set). After this data cleaning, the composition of the resulting datasets is shown in Table 3.

Table 3. Dataset composition after cleaning and unique identifying feature removal.

Dataset	Features	Rows	% Malicious Packets	% Benign Packets
DAPT2020	56	63,775	35	65
SCVIC-APT-2021	60	315,607	1.6	98.4

We created dataset visualizations by using the t-stochastic nearest-neighbor encoding (t-SNE) clustering method on the cleaned dataset on the NPS Hamming supercomputer (Van der Maaten & Hinton, 2008). Figure 1 shows clustering of the DAPT2020 data based on network-traffic features, where colors denote phases. The 40 features were selected from the top scoring features using the "analysis of variance" (ANOVA) F-value (Pedregosa et al., 2011). The SCVIC-APT-2021 authors used similar methods to visualize their dataset (Liu et al., 2022).

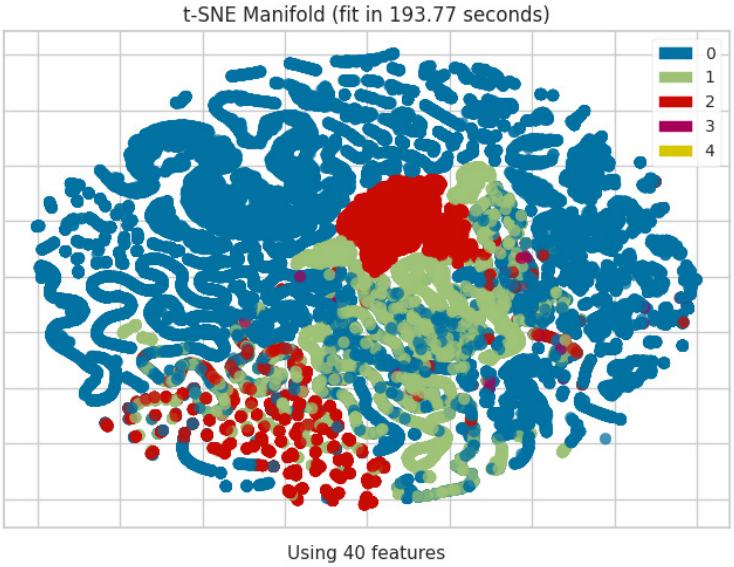

Figure 1. DAPT2020 t-stochastic nearest-neighbor clustering by APT phase using 40 features. This plot is a two-dimensional representation of 40-dimensional space, and is for visualization purposes only (not for quantitative interpretation).

Near-miss undersampling is useful in multiclass classification with classes of widely varying sizes (Bao et. al., 2016). Used with nearest-neighbors clustering, it selects samples from the majority class that have the smallest average distance in the dataset from the three most distant examples in the minority class. This combats bias toward the majority class, which is benign traffic. Classes in the datasets were benign traffic: 0, reconnaissance: 1, establish foothold: 2, lateral movement: 3, data exfiltration: 4, pivoting: 5 (unique to SCIVIC-APT-2021). Near-miss undersampling should reduce both false negatives and false positives by reducing the number of benign traffic samples that are not closely neighboring those of another class (Yen & Lee, 2006). Near-miss undersampling is useful for improving classification of the minority class because it keeps only the hardest cases to classify (those majority class cases closest to the decision boundary).

Attackers try to mask their network traffic by crafting packets that closely match benign packets, making this form of undersampling useful for detection. However, with prioritizing only cases closest to the decision boundary, this method may discard useful samples that are further from the boundary, limiting the ability to generalize the model, such as to the case where there is no malicious traffic (Fernandez et al. 2018). The visualizations of the data after application of near-miss undersampling are shown in Figure 2.

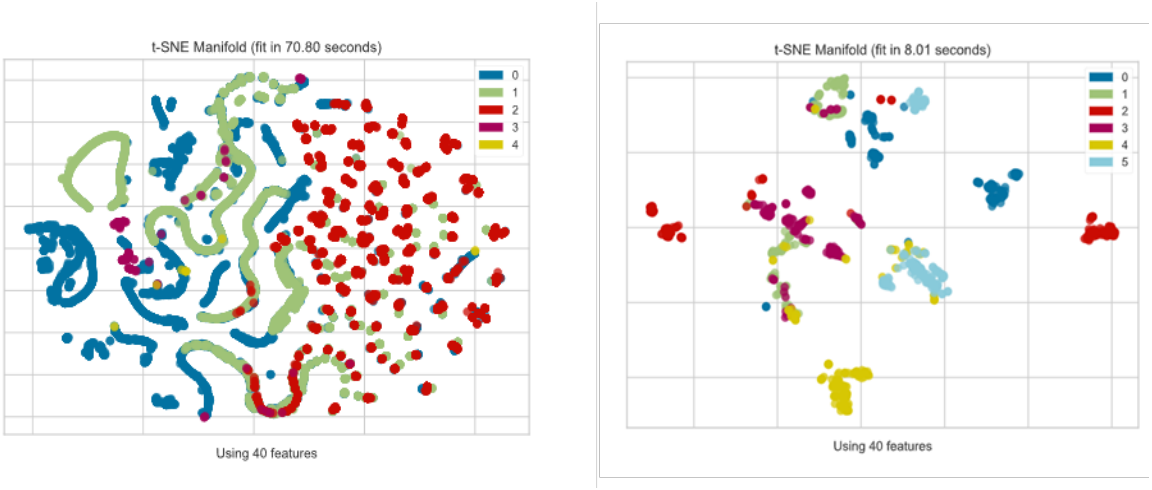

Figure 2. DAPT2020 (left) and SCVIC-APT-2021 (right) datasets visualized by APT phase after cleaning, using t-SNE with near-miss undersampling (version two). This plot is a two-dimensional representation of 40-dimensional space, and is for visualization purposes only (not for quantitative interpretation).

From our exploration of both datasets, we observed that a random-forest classifier could recognize the first few stages of an APT attack with few false negatives, but also produced false positives identifying benign traffic as an APT. The confusion matrices in Figure 3 display the classifier performance on DAPT2020 (left) and SCVIC-APT-2021 (right) without near-miss undersampling.

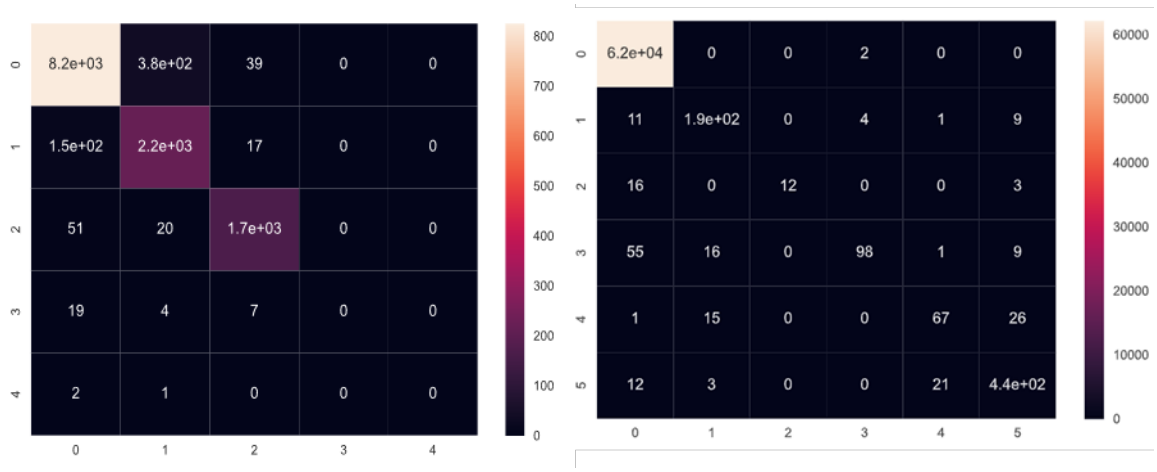

Figure 3. Random forest classifier confusion matrix for DAPT2020 (left) and SCVIC-APT-2021 (right) prior to near-miss undersampling Classes are 0: benign traffic, 1: reconnaissance, 2: establish foothold, 3: lateral movement, 4: data exfiltration, 5: pivoting. The x-axis is the predicted class. The y-axis is the actual class.

Near-miss undersampling improves classification performance in both datasets. The change is especially observable in the SCVIC-APT-2021 dataset, likely due to the low rate of malicious traffic, as near-miss undersampling balances the majority and minority classes when sampling. Figure 4 shows the improvement in confusion matrices after the datasets are undersampled.

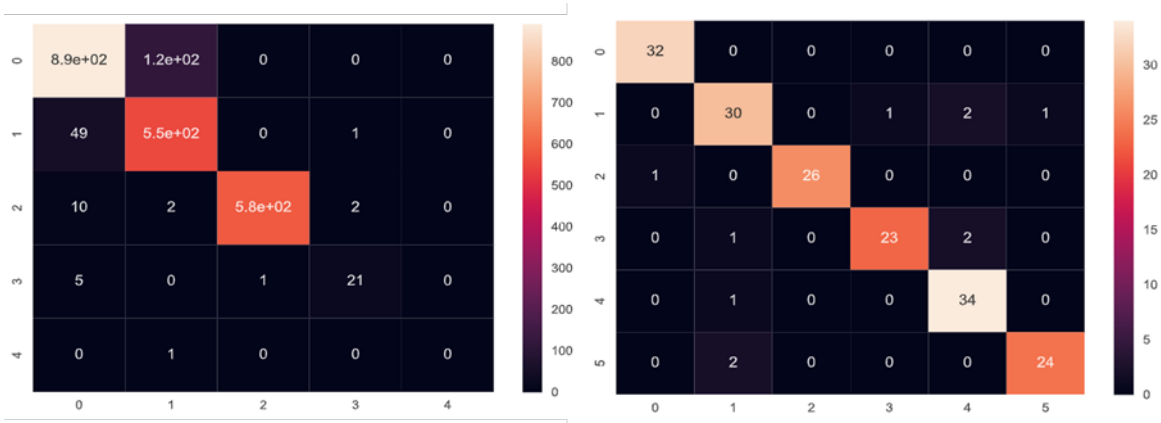

Figure 4. Confusion matrices for DAPT2020 (left) and SCVIC-APT-2021 (right) after applying near-miss undersampling.

Even after near-miss undersampling, many false positives occurred with the DAPT 2020 dataset. The value of near-miss undersampling is not felt as strongly in this dataset, due to the high ratio of malicious to benign traffic when compared with SCVIC-APT-2021. As an added complication, undersampling requires labeled data, which is unlikely to be available for most defenders.

B. CYBER DEFENSE STRATEGIES

Our detection work on DAPT2020 and SCVIC-APT-2021 showed that using machine learning to train a detector to automatically detect an APT and its stage of the attack life cycle is practical. If the defender's goal is not only to gather intelligence, then they must also take actions to prevent and mitigate the damage an APT can cause once it achieves access to a system.

1. Categories of Active Cyber Defense

The basic cyber defense actions (Landsborough et al., 2021) are:

- Deter: Discourage undesirable further actions.
- Deny: Block further actions. This includes actions designed to frustrate or interfere with attacker activities.
- Delay: Slow the attacker.

Deterrence in a cyber context can be passive, such as by securing and building resilient networks, or active, such as through disabling services and threatening retaliation (McKenzie, 2017). Deterrence carries the attributes of credibility, fear, and cost-benefit calculation, which can change rapidly in a confrontation between an attacker and defender.

Deterrence can be by denial or by punishment (Blagden, 2015). Deterrence by denial occurs when the attacker observes costs to outweigh benefits; such deterrence is primarily hardening of systems and strengthening of countermeasures. However, these are unlikely to be effective against APTs, since they often transcend national boundaries, and organizations that sponsor APTs have already decided that benefits outweigh costs. In addition, deterrence relies on the properties of the physical world: that identities are

difficult to hide and that offensive operations have high costs (Fischerkeller & Harknett, 2017). These properties are less likely to be true in cyber space. This removes deterrence by punishment as a credible goal for cyber defense, as attribution and retaliation are ineffective tools for shaping APT behaviors.

Delaying tactics help when the defender wants to learn about the attacker or if the defender is unsure that they are being attacked (Rowe, 2016). Delays may provide time to gain information about tools an adversary uses, what they are interested in, or their identification. A delay may also cause an especially impatient attacker to cease their attack altogether. A noticed delay could suggest to an attacker that they have been detected, deterring those who seek stealth.

Denial tactics can block attacks. They can also create ambiguity or uncertainty for the attacker by concealing facts (Heckman et al., 2015). This includes giving false information to the attacker to impede their understanding of the target system. Providing information for a different type of device than the true one could complicate an attacker's understanding of the system.

The "active cyber defense" strategies of deterring, denying, and delaying can be part of a moving-target defense, in which a defender changes their system to mislead their adversary (Wang & Lu, 2018). Moving-target defense tries to increase the complexity, diversity, and randomness of the protected cyber system, while cyber deception provides plausible but deliberately misleading information to attackers. The two techniques are not mutually exclusive. We now discuss them in more detail.

2. Moving-Target Defenses

When designing a moving-target defense, it is important to consider "what to move, how to move, and when to move" (Cho et al., 2020). "What to move" refers to system or network attributes that can be changed like instruction sets, IP addresses, port numbers, and virtual machines, to modify the "attack surface" or what the attacker sees. "How to move" refers to how a defender can increase unpredictability and uncertainty through shuffling, diversity, and redundancy of the targets themselves such as by protocols, addresses, and software (Xu et al., 2014). "When to move" refers to when to change state,

which can be reactive based on suspicious activity, or proactive based on a schedule or random intervals.

Moving-target defenses try to ensure that whatever information an attacker has gained expires quickly (MacFarland & Shue, 2015). Because of this, they can be useful against APTs because they can invalidate previous attack planning. APTs learn about a system during the reconnaissance phase by exploring networks and identifying targets. This is difficult if the system keeps changing. In a game context, this defense seeks to frequently change the game against an attacker. This makes a moving-target defense effective against an adversary such as an APT that is seeking specific targets (Lei et al., 2018).

Moving-target defenses provide defensive opportunities for legacy systems and existing technologies. However, a defense that is constantly triggered (perhaps due to a probing APT) may hinder services to users, and can be costly to implement, especially if by a third party over a distributed network environment. Also, few trusted infrastructures are secure enough to make moving-target execution decisions.

3. Cyber Deception

Cyber deception can influence an attacker (Ferguson-Walter et al., 2021). Cyber deception can be effective even when the attacker is aware of its possibility if not its techniques. So, it can be useful against APTs because they are likely to continue their attack even with countermeasures and deceptive tactics.

The best-known cyber deception technique is a honeypot, described in Chapter I (Zhang & Thing, 2021). Honeypots can be deployed with several strategies:

- Sacrificial lamb: Denies attackers by having them waste time interacting with a system isolated from the target system.
- Hacker zoo: Denies and delays attackers by having them interact with honeypots of quite varied platforms, services, vulnerabilities, and configurations.

- Minefield: Deters attackers by displaying many obvious fake targets.
- Proximity decoys: Denies attackers by letting them only do a few useless things with the target system.
- Redirection shield: Denies attackers by redirecting them to safer locations or services.
- Deception ports: Delays the attacker by simulating common services such as SMTP, DNS, or FTP on the target system but which allow the defender to collect knowledge about attacker goals.

Cyber-deception techniques such as honeypots must be monitored to stay effective, because attackers can evolve over time. This is especially true against an APT, which can relentlessly pursue its objectives. Repeated attacker engagement with a static defensive system will eventually render the system's ploys ineffective.

Compared to moving-target defenses, cyber deception has the advantages of overhead reduction, understanding of the adversary, and increasing risk for the adversary (Wang & Lu, 2018). Moving targets often require complex system adaptations that have associated computing, network, and hardware costs. Also, moving targets change system boundaries dynamically, and do not try to understand or predetermine adversary attack methods. Cyber deception can engage with the adversary, which allows the defender to learn about the adversary and build better defenses. For the same reason, moving targets are less of a deterrent than cyber deception is to adversaries, because the adversary engagement may deter adversaries such as APTs that do not want to be analyzed.

C. USE CASE

To make our discussion more concrete, we describe a use case of a defender who is using active measures and deception against a hypothetical APT attacker called APT X. This use case does not model any historic APT attack, nor does it represent specific defense measures used by either the U.S. Department of Defense or the private sector, but it shows key principles.

1. Assumptions and Background

We assume that Organization A is a government organization in the defense sector using the software-as-a-service model of cloud computing. This means that Organization A relies on services and applications that run on cloud hardware (Tsai et al., 2014). We also assume that Organization A uses server-side storage provided by the cloud service, so its data resides outside its own network and direct control (Xiao & Xiao, 2013). These cloud services provide essential business functions for Organization A but their administration is governed by a service-level agreement with the cloud provider.

We assume that attack APT X is an APT that receives government sponsorship in exchange for pursuing objectives of national interest for its government such as exfiltration of sensitive defense information, intelligence gathering, and capabilities disruption. As an unofficial and deniable extension of its government, its host government gives it freedom from prosecution and extradition. APT X is known by target organizations like Organization A to seek classified information.

Our use case follows the four APT life-cycle stages used in the DAPT2020 benchmark dataset: Do reconnaissance, establish-a-foothold, move laterally, and exfiltrate data (Alshamrani et al., 2019). The pivot phase referenced in the SCVIC-2021-APT benchmark dataset was excluded because it is not standard to other APT datasets and includes APT activity that can be adequately represented as a separate attack or campaign.

The infrastructure of the system attacked in this use case is shown in Figure 5, and includes Organization A, which uses both internal services and external cloud-hosted services. The cloud provider has a honeypot server, network-monitoring tools, and operates a remote-access portal from which employees of their clients can log in while traveling.

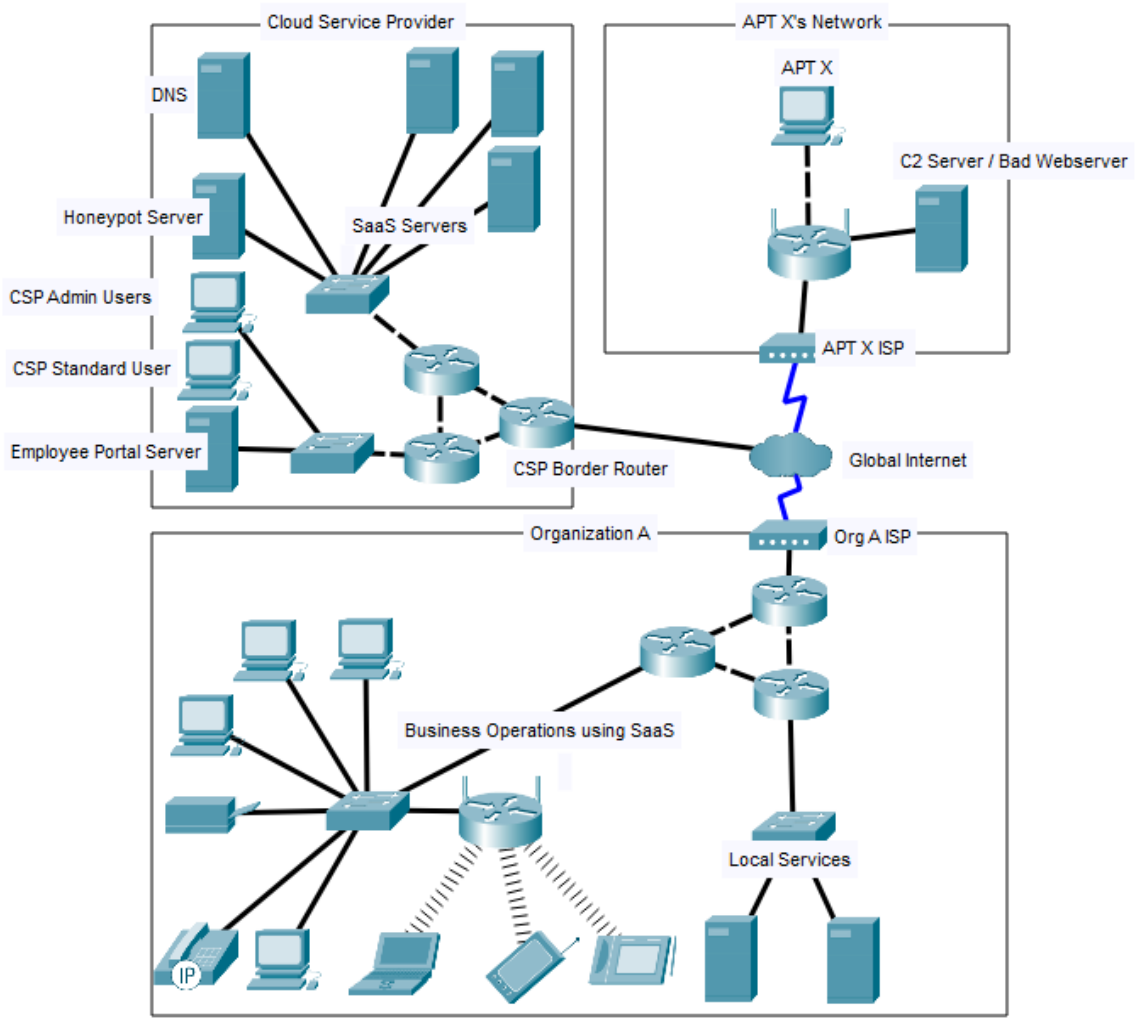

Figure 5. Use case network depiction, built with Packet Tracer (Packet Tracer, version 8.2.0.0162).

2. Reconnaissance Stage

In the reconnaissance stage, APT X seeks information about Organization A, its infrastructure, and its cloud services to choose methods to achieve its objectives (Dargahi et al., 2019). Attackers seek both technical and non-technical information. Technical information is domain names, network topologies, device and operating-system versions, security measures, system processes, host configurations, peripheral devices, application-programming interfaces, databases, account details, and user credentials (Roy et al., 2022).

Non-technical information is organization background, location, logistics, finances, business processes, employee personal information, and contact details.

In the proposed attack, APT X collects employee contact details on the cloud-service provider through a spear-phishing campaign (Winther et al., 2022). Spear phishing is targeted phishing (online scamming) (Wang et al. 2012). APT X can get the necessary contact information from social media, employment websites, and other open information sources. Using social engineering, the targets of the campaign are sent an email that looks legitimate and urgent and instructs them to follow a link that shows them a facsimile of the cloud provider's employee portal at "www.employeelopin.csp.com" instead of "www.employeelogin.csp.com." Employees enter their credentials there, and the page responds with "Invalid Credentials" but records their login entry. The page then refreshes, redirecting and presenting the actual cloud employee portal. This is shown in Figure 6, with APT X in red, and a victim employee of the cloud-service provider in green.

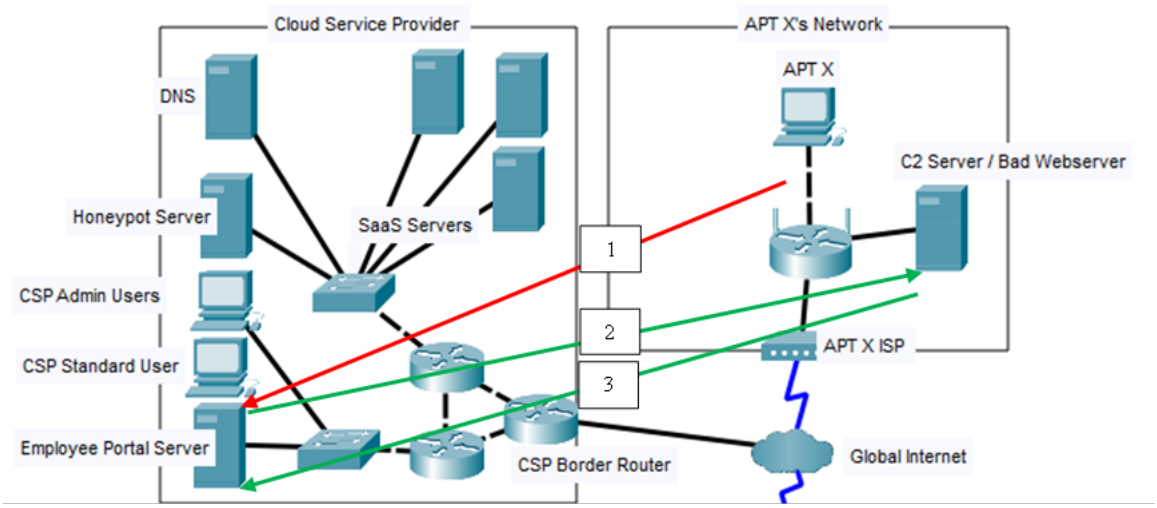

Figure 6. Reconnaissance phase of APT. Action 1: spear phishing campaign. 2: Log in to simulation of the employee login page, hosted by APT. 3: Log in on actual login portal.

APT X can use network-reconnaissance tools to scan the target network to determine services available at particular IP addresses on particular ports. It can disguise itself by using the SYN flag in the TCP header (Bhuyan et al., 2011). It may also use other

flags in the TCP header (FIN, ACK, etc.) to do a "half-open scan", simply seeking a response from the target to see whether the port specified is listening or not, without completing the connection (Panjwani et. al., 2005).

For the defender, their objective during reconnaissance should be to discourage the attacker from gaining actionable information about its systems by presenting a hardened target. This could mean that the attacker's reconnaissance takes longer than normal, that they end their reconnaissance early because of the increasing work required to gain information, or that they receive false information that impedes attack development.

Getting authorized-user credentials on the system is desirable for an APT because this lets them move quietly through the target system while pretending to be a legitimate user. The APT can log on like a normal user and explore the target system with less chance of alerting defenders to its presence. To increase the difficulty of getting authorized-user credentials, the cloud provider could use an active defense such as multi-factor authentication.

To thwart scanning by an APT, assuming address and port scanning are known to network-defense software, the targeted devices could return incorrect information (Birkinshaw et al., 2019). Knowing it is targeted by reconnaissance, the server could respond positively with a SYN-ACK flag to a SYN flagged request from the attacker on ports that are not actually providing services. On the other ports, the server could respond with the RST flag for the first few requests, claiming to not host the service associated with that port. An attacker that receives false information, requiring them to target systems many times before they succeed in connecting, is more likely to be detected. This can raise attacker frustration as they receive conflicting information from the target systems. It could also frustrate legitimate users, although we can warn them beforehand.

3. Establishing-a-Foothold Stage

The APT establishes a foothold to exploit vulnerabilities it found during reconnaissance (Alshamrani et al., 2019). The APT may run internal reconnaissance to further survey the target network and locate targets, stage subsequent attacks, launch malware to compromise and provide it control of network devices and hosts, and establish

command-and-control channels that enable long-term access to target devices (Lehto, 2022).

Figure 7 shows a situation in which APT X has obtained valid user credentials from the mirrored website and logged onto the system. It then downloads containerized malware and reconnaissance tools, evading intrusion-prevention systems if the container lacks known malware signatures (Dietrich et al., 2011). APT X extends its access by installing persistent backdoors and establishes remote command-and-control channels connecting to its own systems by protocols like DNS.

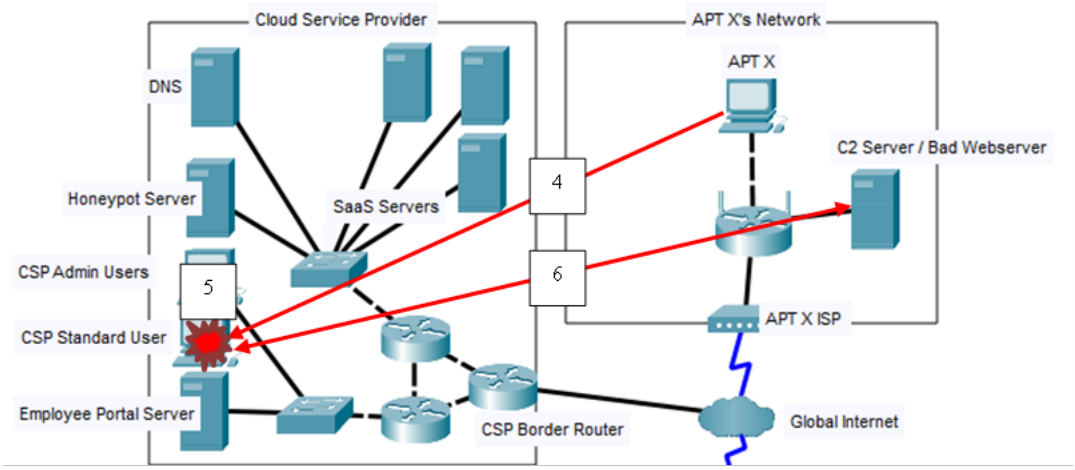

Figure 7. Establishing a foothold. Action 4: Logs into system using user-level credentials. 5: Installs containerized malware. 6: Establishes C2 channel back to its C2 server.

The defender's goal for thwarting this stage should be to improve the chances of attack detection by impeding it so that it generates larger signatures. This gains time for the defender to marshal a response, and helps the defender recognize targets. The defender can impede the APT by having the system make additional demands on the APT, giving false information about available services to them, and issuing false warnings to them about updates that will reset systems. These interactions could delay the APT considerably.

These effects can be accomplished with honeypots (Chapter II.B.3) and honeytokens (Lackner, 2021). Honeytokens are data or files that alert defenders and trigger

automated responses when accessed (Bourke & Grzelak, 2018). An example is a fake network address which alerts the defender to the presence of an attacker. A honeytoken may cause a session to be isolated or terminated, provide the attacker with false information about systems, impose connection timeouts, display additional vulnerabilities to interest an attacker, or encourage attackers to establish connections that could make it easier to track them.

4. Lateral-Movement Stage

With lateral movement, an APT explores a network, locating potential targets and high-value nodes such as network infrastructure, data storage, or industrial controls (Maicon et al., 2016). Lateral movement may also expand access to the network. Often the APT seeks administrator credentials to access highly protected resources or deploy further malicious software (Oliveira et al., 2022). APTs often use containerized software at this stage because it complicates detection and administrators do deploy legitimate containers.

Persistence of access is a major goal for APT X at this stage, which it can achieve by creating new users, giving them remote-login capabilities, and adding their processes to startup profiles (Li et al., 2022). APT X can edit logon scripts on the network and hosts to do this (MITRE, 2022). Once persistence is established, the APT can use tools against the servers in the cloud for easier reconnaissance (Ibrahima et al., 2022). The APT can also access portals to capture customer information, credentials, and access tokens (MITRE, 2020) and other sensitive data. As it moves through the network, APT X can use the administrator credentials of the compromised user machine in a “Pass-the-Hash” attack (Amin et al., 2021). This allows it to create new user accounts on the cloud provider (Figure 8).

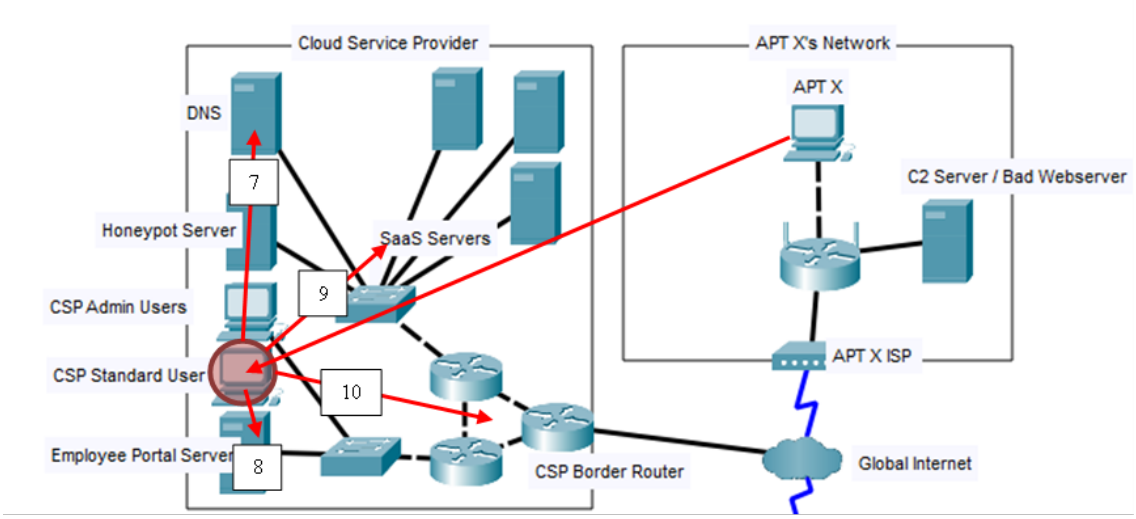

Figure 8. Lateral movement through the target network. Action 7: "Pass the hash" to gain administrator credentials. 8: New account creation. 9: Enter SaaS servers using administrator access. 10: Enable persistent access with login script editing on network devices.

The defender's goal during the APT's lateral movement should be to delay the attack. Tactics could direct the attacker toward honeypots and honeynets (Lackner, 2021). Systems could mislead the attack by randomly reporting that they are doing things like powering down, logging the APT's actions as suspicious, receiving updates, or are mandating a password change, to create more problems for the APT to solve.

5. Data-Exfiltration Stage

Once APT X has the desired access to its target, it can exfiltrate data from it. It may use diversions to confuse analysis, such as a distributed denial-of-service attack, to camouflage exfiltration. It can also conceal its activity by modifying logs, clearing command histories, corrupting system files, and erasing storage (Strom et al., 2022) (Figure 9).

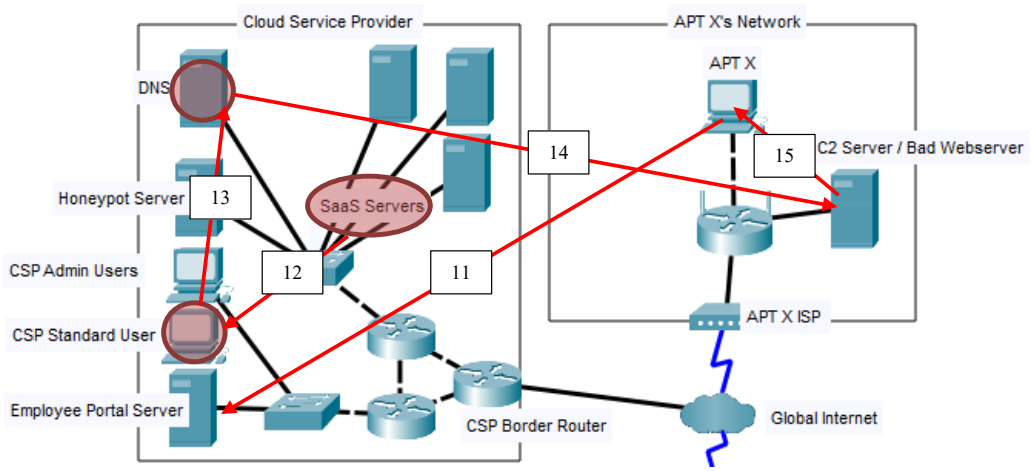

Figure 9. Exfiltrating data. Action 11: Denial of service on the employee portal server. 12: Reading data. 13: Encoding data into timing channel with DNS requests. 14: DNS requests to APT's server. 15: Data is decoded by APT X.

The APT could exfiltrate data as attachments to email (Ullah et al., 2018). As attachments are common, this will not trigger alarms, and afterwards the email could be quickly removed from the mail servers. Another exfiltration method could briefly post data to public websites for download. Another option is a timing channel, sending innocuous packets so that the time delay between packets encodes a byte value. The DNS and ICMP protocols are good for this because they are used frequently for normal network operations.

The defender during this phase should impede exfiltration. They can give the APT information that is incorrect, intentionally corrupted, or encrypted (perhaps in response to an alert) by providing false reports of unavailability, requiring many tries to access data, or increasing latency by decreasing the maximum packet size permitted on the network. A countermeasure for timing channels is an encrypted mix-network (Ullah et al., 2018) which randomly changes the destination addresses for packets bound for the same destination. This would require the observer collecting the outbound packets to assemble the randomly addressed packets.

Data exfiltration can be detected by fingerprinting each packet with identifying details of who sent it (Ullah et al., 2018). To improve this, a second decoy document could be created with incorrect information for each file exfiltrated. Both documents could be stored on the cloud, and when a suspicious user tries to access the file with correct information, they are provided with the decoy. Intrusion-detection software can detect data exfiltration, but such software must be concealed. It could analyze packets through a separate device that ("bump-in-the-wire") that is invisible to scanning (Cojocar et al., 2021). Hiding intrusion-detection systems is important to prevent attackers from deactivating them.

6. Use Case Discussion

This use case shows that many options are available to the attacker, but also many countermeasures are available to the defender. A full defense should include not only passive measures such as firewalls, intrusion-detection systems, and intrusion-event management systems, but also active measures to deter, delay, and deny adversary actions. Such measures could be deployed when the defender's system recognizes that it is under attack, by using specialized hardware or software designed to trigger at APT attack-lifecycle clues.

Developing a plan for when, where, and why these techniques are deployed is important to their effective use. It could allow defenders to automatically coordinate the active defense to match attacker techniques. This plan must be comprehensive, including the APT's choice of attack vector and their likely targets, and must store data of previous interactions with the APT. Game theory could help in developing it because of the many possible attack-defender interactions.

IV. GAME-MODELING METHODOLOGY

APTs can be modeled as a multi-stage game (Zhu and Rass, 2018). Detection of the APT is a critical goal for the defender. APTs use so many tactics to remain stealthy and persistent that they are best identified by automated machine learning. APT actions can most easily be distinguished from benign traffic during reconnaissance and establishing a foothold, as was seen during our phase classification experiments of Chapter III.A.2. Otherwise, the interactions between the APT and defender provide clues to the APT's goals, tactics, techniques, and procedures.

A. GAME DESIGN

Our game design has the standard components of players, actions for each player, preferences among actions for each player, and moves made sequentially by alternate players (Osborne, 2004). Our players were a simulated APT attacker and a simulated cybersecurity group defending a local-area network. We encoded the actions for each player from APT tactics and available defender actions in the MITRE ATT&CK framework (MITRE, 2023b). Preferences for actions for each player were determined by action costs, benefits, probabilities of success, and duration required, which we specified. Cost is mainly determined by duration, but additional factors such as intellectual difficulty were also included.

1. Player Generation

For testing, we modeled a diverse group of APTs: APT39/Remix Kitten, Lazarus Group, MuddyWater, Sandworm Team, Turla, Wizard Spider, APT31/Zirconium, and our main use case of APT X. Table 4 shows their key features.

Table 4. APTs that provide attacker actions for experiments.

APT	Suspected Attribution	Goals	Targets	Preferred Tactics
APT39/ Remix Kitten (Hawley et al., 2019)	Iran	Sensitive data exfiltration, political repression	Political targets, including foreign dissidents	Connecting to victim machines remotely for persistence and lateral movement, data exfiltration uses zip files, custom tools
Lazarus Group (Park, 2021)	North Korea	Data exfiltration, intelligence gathering, sabotage, financial gain	Financial institutions, government agencies, entertainment industry	Holding critical data hostage in exchange for Bitcoin ransom
MuddyWater (Avertium, 2022)	Iran	Intelligence gathering, financial gain, sensitive data exfiltration	Government and private sector defense, energy, government, and telecommunications industries	Using open-source tools
Sandworm Team (Cunningham, 2020)	Russia	Cyber sabotage	Critical infrastructure, particularly energy-related	Long-term persistence
Turla (Faou, 2019)	Russia	Intelligence gathering and cyber sabotage	Geopolitical adversaries, international organizations	Custom tools and malware, obscuring destination of exfiltrated data
Wizard Spider (DiMaggio, 2021)	Russia	Financial gain	Banking institutions	TrickBot Trojan ransomware, exploiting Wake-on-LAN capability to spread
APT31/ Zirconium (Soesanto, 2021)	China	Intelligence gathering	High-level US and international community election campaign personnel	Collecting data about Web browsing (Fonseca et al., 2005), repurposing exploits from other APTs
APT X/Use Case (Ch 3.c)	Generic	Sensitive data exfiltration	Cloud-service providers	Trusted third party compromise

Table 5 shows the key attributes of these APTs used in the simulation, based on historical observations:

- Resources: How well-resourced an APT is on a scale from 1-3 (1 = non-state, 2 = state-sponsored, 3 = state). Considerations are available tools, available skills, intelligence support, and financing.

- Stealth: The level of concern that an APT has for remaining undetected on a scale from 1-3 (1 = low concern, 2 = medium concern, 3 = high concern). Considerations are reputational and retaliatory concerns, and potential for sanctions and political fallout associated with being attributed.

- Preferred tactics: Tactics that an APT has used historically, derived from the descriptions in the MITRE ATT&CK enterprise layer. This is done by matching tactics from the layer to their action profile. How these tactics are used is described in Chapter IV.A.2.

- Preference weight: The APT's historical preference for their known attack methods in the preferred tactics set (1: no preference, 0.8: moderate preference, and 0.6: strong preference). Attackers with strong preference repeatedly use the same techniques.

Table 5. APT attributes based on historical observations. Resources, stealth, and preference weight are described in IV.A.1.

APT (Abbreviation)	Resources	Stealth	Preference Weight
APT39/Remix Kitten (RK)	3	1	1
Lazarus Group (LG)	2	2	0.8
MuddyWater (MW)	2	3	0.6
Sandworm Team (SWT)	3	1	0.8
Turla (Tur)	2	2	0.6
Wizard Spider (WS)	1	2	0.6
APT31/Zirconium (Zirc)	2	3	0.8
APT X/Use Case (UC)	3	3	1

The defender could be a commercial or public entity such as a cloud-service provider, military-network administrator, or critical-infrastructure cybersecurity team. The defender attributes in the simulation were:

- Resources available: How well-resourced a defender is on a scale from 1-3 (1 = few resources available for cybersecurity, 2 = medium amount of resources, 3 = many resources).

- Confidentiality, integrity, and availability importance: The defender's tolerance for risk for each of the basic information security principles (Fenrich, 2008). For example, one organization may be concerned about data compromise (higher concern for confidentiality), whereas another may only be concerned about maintaining operations (higher concern for availability). We used values of 3, 2, or 1, with 3 being the least concern, and 1 being the highest concern. The multiplier applied for highest priority is 1.25, the second is 1.15, and the third is 1.

- Initial security state: A list of attributes of the environment before the game begins (see Appendix A for the full set of possibilities). They are either hardening attributes or vulnerabilities. Hardening attributes represent defender preparation such as security policies and installed defense systems; examples are firewalls and strong passwords. Vulnerabilities are flaws in the target system such as an unobservant user and software bugs. Our experiments used four security states:
    - Very low security: 10 vulnerabilities, 0 hardening
    - Low security: 10 vulnerabilities, 6 hardening
    - Medium security: 5 vulnerabilities, 13 hardening
    - High security: 0 vulnerabilities, 19 hardening

2. Action Profiles and Player Specification Generation

For our experiments, 65 action profiles were generated for the attacker and 78 for the defender; Angela Tan created additional defender profiles that were modified to fit our framework. We primarily focused on techniques in which the attacker seeks technical instead of social information about a system, because it can be difficult to defend against all the methods of the latter. Sample player specifications are shown in Appendix A. Each action profile had these attributes:

- Phase. The APT life-cycle phase in the DAPT2020 dataset in which an action is done.
- Action Name. The action in the MITRE ATT&CK Framework v13 (MITRE, 2023b), a MITRE repository of cybersecurity attacks and countermeasures (Kaloroumakis & Smith, 2021), the Nmap Reference Guide (Nmap Project, n.d.), a survey on adversarial reconnaissance techniques (Roy et al., 2022), or a review of attack vectors and countermeasures (Ullah et al., 2018). If there was overlap of actions between the sources, we defaulted to using a generalized action name.

- Action Class. These were:
    - Attacker, Phase 1: scanning, sniffing/spoofing/observing, host-based reconnaissance, third-party reconnaissance, and human-based reconnaissance.
    - Attacker, Phase 2: system-based actions, human-based actions, and execution.
    - Attacker, Phase 3: evasion, privilege escalation, and obtaining persistence.
    - Attacker, Phase 4: active and passive.
    - Defender, All Phases: moving-target defense, cyber deception, deception, and hardening.
- Cost. The cost to execute an action was estimated based on its technical sophistication (*S*), resources available to a player from IV.A.1 (*R*), the "noise level" of an action based on observations from III.A (*N*), a player's concern for remaining stealthy from IV.A.1 (*C*), and a weight on the action. Noise level is defined as the degree of generation of statistical anomalies or artifacts that provide an indicator of compromise to a defender (Hare & Diehl, 2020). For attacker action, the weight was the preference-weight attribute from IV.A.1 representing an APT's observed historical preference for it, or for the defender, a cost adjustment for action class (hardening and detection were given a 1, moving-target defenses a 2, and cyber deception a 3). The defender weights for hardening and detection were assumed to follow an organization security policy and therefore have low costs; cyber deception required credibility and interaction to deceive adversaries. Cost was determined by:

$$\text{Cost} = \left(\left(\frac{S_{action}}{R}\right) + (N * C)\right) * W$$

$$N_{defender} = 0$$

$$C_{defender} = 0$$

$$W_{hardening} = 1$$

$$W_{detection} = 1$$

$$W_{mtd} = 2$$

$$W_{deception} = 3$$

- Probability of success. Values were high (default 95%), medium (45%), low (15%), or very low (1%). Actions of significant complexity or reliant on defender negligence had lower probabilities of success.
- Benefit. This is based on the values of postconditions after successfully doing the action. For example, a high benefit was assigned to an attacker successfully exfiltrating data, and to a defender successfully luring an APT to a honeypot to study it.
- Duration. One part of the cost of an action is proportional to its duration. Duration is incorporated by multiplying the stated duration by a delay multiplier (0.05 was used for our experiments) and adding the result to the cost.
- Preconditions. These are necessary conditions for an action to occur, expressed as facts.
- Negative preconditions. These are conditions that must be false for an action to occur.

- Postconditions. These conditions become true (if not already) after a successful action performance. Postconditions can be negative to represent facts that become false after the action. For our model, benefits on a scale from 1-5 were tied to postconditions.

- Support goal. These were the general purpose of an action for the player, which for attackers were discovery, network observation, user redirection, cryptosystem analysis, vulnerability enumeration, footprinting, obtaining credentials, initial compromise, execution of a payload, evading of defenses, obtaining of persistence, data exfiltration, and cleanup. Defender support goals were denying, delaying, deterring (the three categories of active cyber defense), and detecting.

Visualizations were produced to help understand how actions relate across the four phases. Partial attacker and defender visualizations are shown in Figures 10 and 11. These show some of the possible paths between phases of the game.

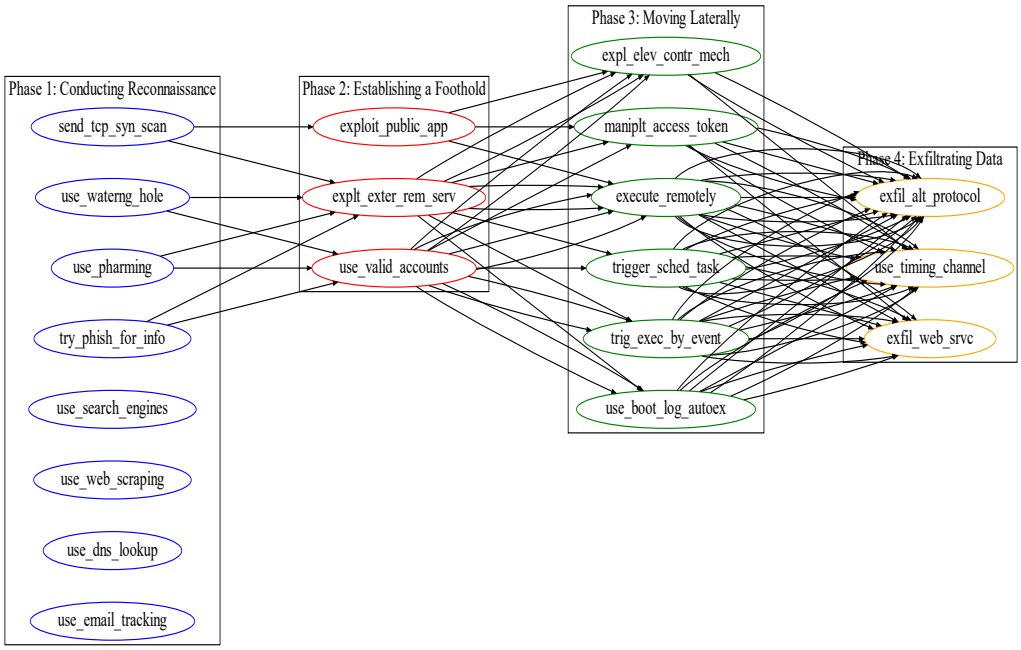

Figure 10. Partial subset of APT X's attack profile, represented as a directed graph of its preferred actions.

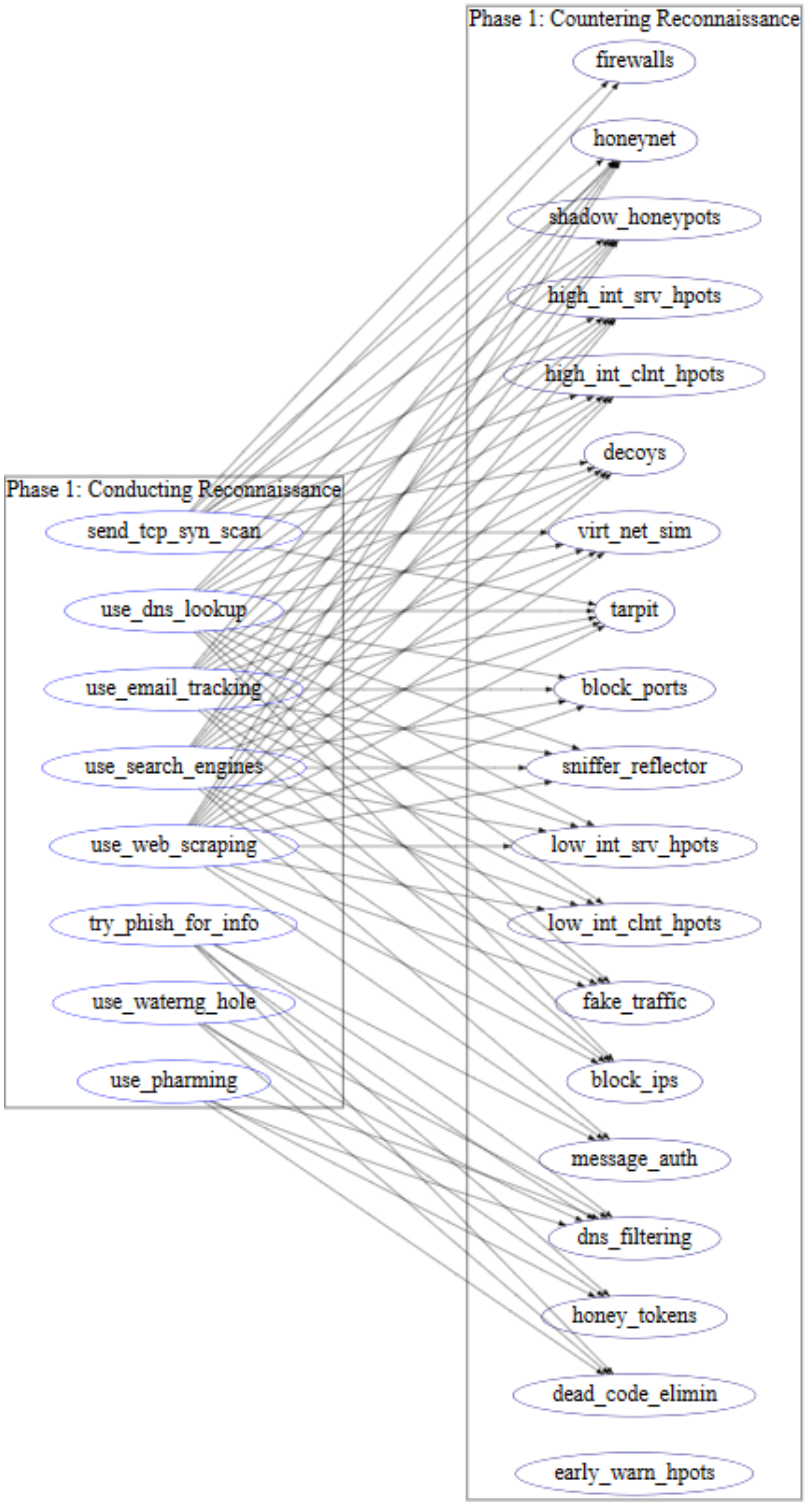

Figure 11. Partial defense subset for the use case cloud-service provider defender for Phase 1, represented as a directed graph.

B. IMPLEMENTATION METHODOLOGY

The player profiles were inputs to a game modeling program written by Prof. Neil Rowe. This program runs games with the attacker and defender profiles from a specified initial state. It scores each action over the series of games and uses reinforcement learning to choose actions with better scores with higher probabilities. Actions with more positive scores are better for the defender, and actions with more negative scores are better for the attacker. Besides the profiles and initial state, other parameters in the game program can be set:

- Maximum moves. This is the number of moves played per game by both players alternating with the attacker starting. Values used were from 50-300 moves per game.

- Number of games. This is the number of games that the attacker and defender play, updating the reinforcement on actions at the end of each game. For our experiments, this was set to 100.

Mean (c) and slope (s) of the sigmoid logistic function were used for calculating reinforcement. The logistic function is defined as:

$$f(x) = 1/(1 + e)^{-s(x-c)}$$

This maps from an average score over games using a particular action to a probability of selecting that action in a future game. Probabilistic selection is important so that the players do not become too predictable, but the actions that have been seen to be better should be selected more often. The program randomly selects actions for each move in the game, among the actions that are permitted by their preconditions. Actions may fail with a specified probability; in which case they incur costs but no benefits. The final score for a game is defined by an evaluation function:

$$\text{Score} = \text{Benefit}_{\text{defender}} - \text{Cost}_{\text{defender}} - \text{Benefit}_{\text{attacker}} + \text{Cost}_{\text{attacker}}$$

With variations in priority ordering of confidentiality, integrity, and availability, and the initial environment security states of high, medium, low, and very low, 72 kinds of defenders are possible. Table 6 shows the 13 representative combinations we tested.

Table 6. Defender specifications used for testing. Row descriptions were described in Chapter IV.A.1.

Defender (Abbreviation)	Resources	Priority of Confidentiality (C), Integrity (I), and Availability (A)	Initial Security State
Alpha (A)	1	(C, I, A)	High
Bravo (B)	2	(C, I, A)	High
Charlie (C)	3	(C, I, A)	High
Delta (D)	1	(I, C, A)	Medium
Echo (E)	2	(I, C, A)	Medium
Foxtrot (F)	3	(I, C, A)	Medium
Golf (G)	1	(A, I, C)	Low
Hotel (H)	2	(A, I, C)	Low
India (I)	3	(A, I, C)	Low
Juliet (J)	1	(C, I, A)	Very low
Kilo (K)	1	(C, I, A)	Very Low
Lima (L)	1	(I, C, A)	Very Low
Mike (M)	1	(A, I, C)	Very Low

Games were played for each defender in Table 6 against each attacker in Table 4, while keeping the sigmoid parameters fixed. We varied the number of maximum moves per game from 50 to 300 with increments of 5. We then looked for indications that reinforcement learning helped increase defender performance across the 13 specifications.

THIS PAGE INTENTIONALLY LEFT BLANK

V. RESULTS AND DISCUSSION

A. RESULTS

A total of 5,304 tests were done of 104 attacker-defender pairs. Each test played 100 games, with 50-300 allowed turns per game in increments of 5 turns. Statistics were collected for the average final scores (Table 7). In general, the final scores initially increased with number of allowed turns as learning occurred, but a maximum score average occurred where the attacker started to overcome the defenses. Statistics were collected for the average number of turns it took to achieve this maximum score (Table 8).

Table 7. Average final scores of 104 attacker-defender pairs over 5,304 tests (for 100 games per row and 50-300 turns per game incremented by 5). Column headings are APT abbreviations from Table 5. Row headings are defender abbreviations from Table 6. Def Avg Score is the average of the final scores in the row (average performance of the defender). APT Avg Score is the average of the final scores in the column (average performance of the attacker).

Defender	RK	LG	MW	SWT	Tur	WS	Zirc	UC	Def Avg Score
A	-2.47	32.73	153.72	-29.93	18.98	83.26	173.11	189.63	77.38
B	41.63	149.51	279.75	38.28	114.08	179.95	287.05	236.57	165.85
C	58.16	194.13	247.86	2.70	156.54	174.21	301.05	296.47	178.89
D	3.22	38.69	113.30	-28.80	20.63	85.48	170.22	117.33	65.01
E	44.95	144.51	247.56	43.79	100.09	169.47	232.07	235.78	152.28
F	20.58	136.68	257.82	29.80	110.03	188.66	301.35	292.14	167.13
G	-60.32	14.86	126.63	-37.10	4.23	53.85	109.07	82.98	36.78
H	6.60	94.04	204.70	-38.81	72.91	87.96	269.60	190.17	110.90
I	52.45	100.36	293.36	33.03	81.97	137.06	338.82	213.17	156.28
J	-55.20	30.24	239.55	-34.18	63.09	162.08	347.47	331.48	135.57
K	18.50	36.03	326.91	-9.01	94.36	41.14	446.78	260.50	151.90
L	-34.28	93.12	240.82	-0.17	45.04	56.54	195.06	209.92	100.76
M	26.30	-6.70	54.90	-55.15	26.40	56.21	118.34	137.17	44.68
APT Avg Score	9.24	81.40	214.38	-6.58	69.87	113.53	253.08	214.87	

Table 8. Average number of turns at which the defender maximizes score over 5,304 tests (with 100 games, 50-300 turns per game, incremented by 5). Column and row headings are the same as Table 7. Avg Turn is the average of the number of turns in a game series at which the defender maximized its score. Averages are rounded to the nearest turn.

Defender	RK	LG	MW	SWT	Tur	WS	Zirc	UC	Avg Turn
A	290	230	255	130	285	295	280	275	255
B	235	285	230	120	265	280	280	180	234
C	235	265	300	145	250	285	285	275	255
D	260	240	290	215	280	220	280	125	239
E	250	275	275	240	275	270	250	260	262
F	290	250	215	230	260	235	280	300	258
G	270	225	150	180	230	140	215	250	208
H	260	230	300	275	285	245	290	255	268
I	255	225	295	175	270	145	295	245	238
J	255	285	300	195	280	250	295	300	270
K	245	235	295	195	240	250	285	280	253
L	275	285	300	240	180	220	295	295	261
M	275	135	250	205	285	270	270	285	247

Because Table 7 only shows average final scores as the number of total turns is incremented, plots were also produced to show the trend of the final game score as the maximum number of turns increased over an interaction. Figures 12 and 13 show examples of these plots for two defenders with varying initial security states and amounts of resources, against the MuddyWater APT. Attacker-defender interactions with a higher security state environment generally follow the trendline of Figure 12, with final scores increasing as the defender learned how to counter the attack, but a maximum where the attacker learning began to decrease the final score. Attacker-defender interactions in a lower security state environment generally follow the trendline of Figure 13, where the maximum defender score was early on, with a rapid drop as games lengthened.

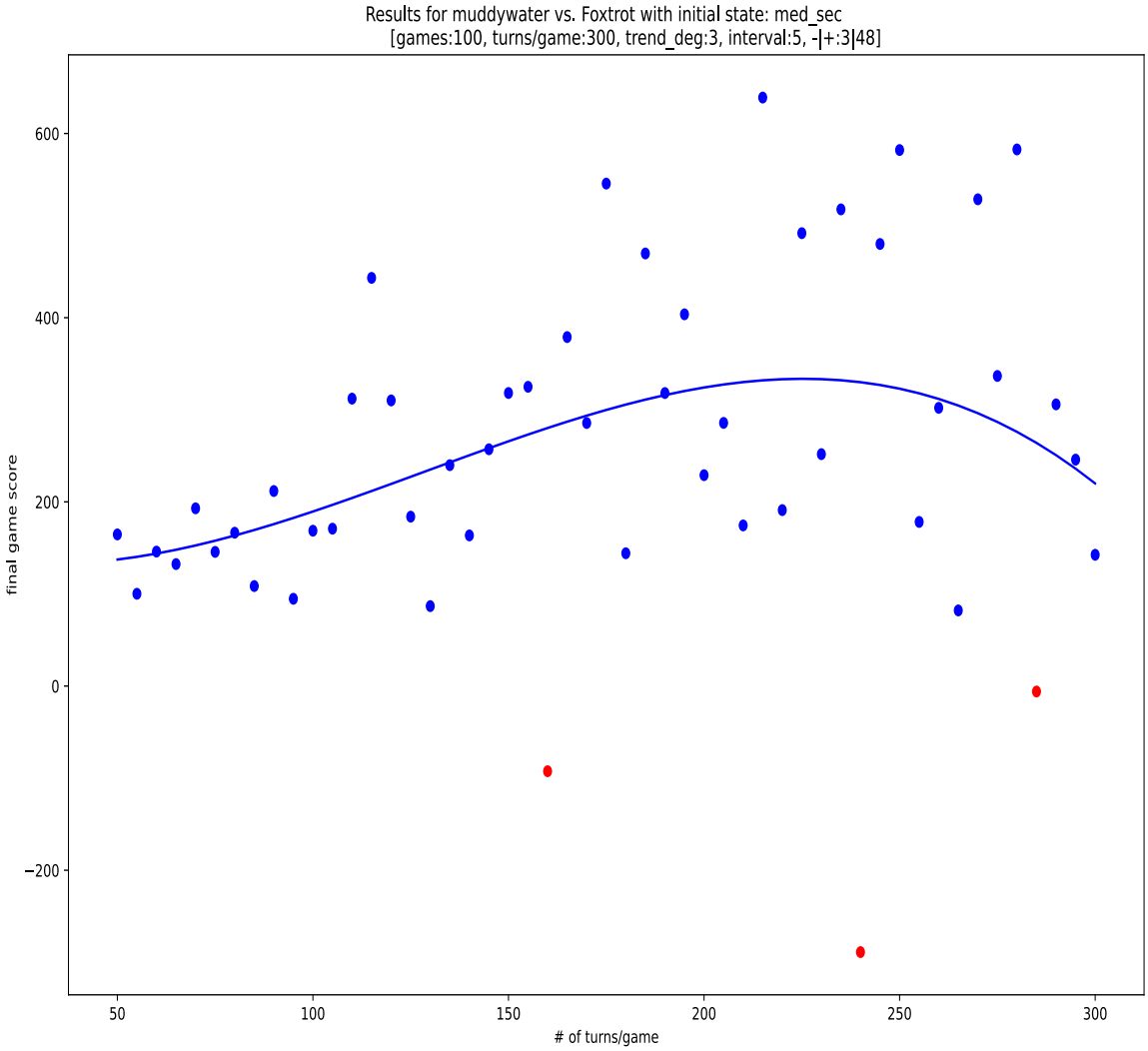

Figure 12. MuddyWater APT versus Foxtrot defender (a medium security defender with high resources). The horizontal axis represents the number of turns per game for each of the 100 games played. The vertical axis represents the final score of each 100-game interaction. Blue dots indicate a positive final game score. Red dots indicate a negative final game score. The trendline shows a 3rd degree polynomial fit.

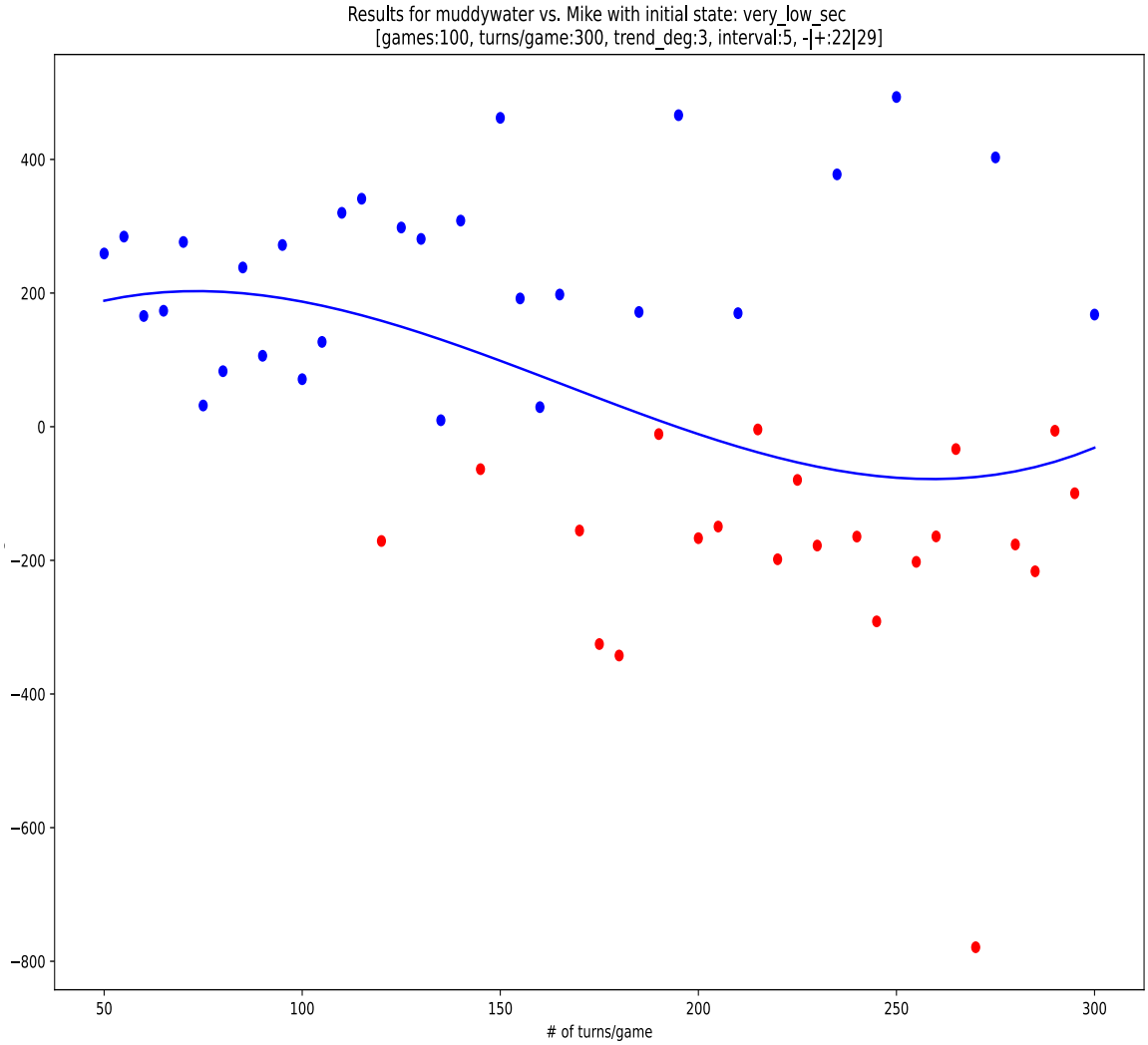

Figure 13. MuddyWater APT versus Mike defender (a very low security defender with the least resources). The horizontal axis represents the number of turns per game for each of the 100 games played. The vertical axis represents the final score of each 100-game interaction. Blue dots indicate a positive final game score. Red dots indicate a negative final game score. The trendline shows a 3rd degree polynomial fit.

Statistics were also collected on the defender actions overall across all games by phase (Table 9). Some actions scored well relative to other actions in the same phase but did not score well when compared to actions in other phases. This distinction is shown in Table 10. The game sequence is likely to affect this, as the later phases are not reached as often as early phases, so more data was collected about actions in earlier phases.

Table 9. 10 highest-scoring (shaded gray) and 5 lowest-scoring defender actions (across 5,304 tests played with 8 attackers, 13 defenders, and 100 games per test, 50-300 turns per game). Phase is the APT life-cycle phase (reconnaissance, establishing-a-foothold, lateral movement, or data exfiltration).

Action	Action Class	Action Score	Phase
deploy_honeynet	Deception	30817.7	1
use_high_int_clnt_hpots	Deception	29459.2	2
create_decoy_public_release	Deception	29190.8	1
deploy_honey_tokens	Deception	29117.4	2
create_decoy_file	Deception	29077.9	1
use_shadow_hpots	Deception	28980	3
harden_sys_config_perm	Hardening	28905.1	3
create_decoy_website	Deception	28903.3	1
check_excp_hand_pntr	Deception	28799.1	2
create_decoy_session_cred	Deception	28770.3	1
isolate_kernel_proc	Hardening	5002.72	2
authorization_event_thresholding	Hardening	4944.9	1
use_biometric_auth	Hardening	4899.64	1
authentication_event_thresholding	Hardening	4825.74	1
use_proc_seg_exec_prev	Hardening	4794.01	2

Table 10. Highest-scoring (shaded gray) and lowest-scoring defender actions over 5,304 tests, by phase. Some actions within a phase are both highest and lowest-scoring because only a few actions were chosen by the defender for the phase.

Action	Action Class	Score	Phase
deploy_honeynet	Deception	30817.7	1
create_decoy_public_release	Deception	29190.8	1
create_decoy_file	Deception	29077.9	1
create_decoy_website	Deception	28903.3	1
create_decoy_session_cred	Deception	28770.3	1
authent_msgs	Hardening	8535.44	1
encrypt_msgs	Hardening	8482.55	1
authorization_event_thresholding	Detection	4944.9	1
use_biometric_auth	Hardening	4899.64	1
authentication_event_thresholding	Detection	4825.74	1
use_high_int_clnt_hpots	Deception	29459.2	2
deploy_honey_tokens	Deception	29117.4	2
check_excp_hand_pntr	Hardening	28799.1	2
use_seg_addr_offset_rand	Moving-target defense	28662.7	2
analyze_system_call	Detection	28567.6	2
auth_pointer	Hardening	28541.3	2
harden_user_acc_perm	Hardening	28094.8	2
restrict_io_port	Hardening	8913.76	2
isolate_kernel_proc	Hardening	5002.72	2
use_proc_seg_exec_prev	Hardening	4794.01	2
use_shadow_hpots	Deception	28980	3
harden_sys_config_perm	Hardening	28905.1	3
isolate_bcast_domain	Hardening	14289.2	3
bootloader_auth	Hardening	8664.57	3
drvr_load_integ_chck	Hardening	8456.74	3
isolate_bcast_domain	Hardening	14289.2	3
bootloader_auth	Hardening	8664.57	3
drvr_load_integ_chck	Hardening	8456.74	3
isolate_hw_proc	Hardening	5096	3
lock_accts_on_sched	Hardening	5077.07	3
encrypt_file	Hardening	8641.03	4
encrypt_disk	Hardening	8624.66	4
encrypt_file	Hardening	8641.03	4
encrypt_disk	Hardening	8624.66	4

B. DISCUSSION

Reinforcement learning improved defender scores as shown in Figures 12 and 13, where defender scores improved as turns increased. The attacker was seen learning as well, as the attacker sometimes minimized the game score in the second third of the interaction (150-250 turns). Both the attacker and defender could improve as the games lengthened, but the scale of these improvements were unpredictable.

Some of our subsequent analysis used a t-test for statistical significance of particular values of game parameters in impacting game outcomes (Hsu & Lachenbruch, 2014). Formula used for t was:

$$t = \frac{\bar{x} - \mu}{s\sqrt{n}}$$

The t-value was then compared using a one tailed distribution t-table to resolve its corresponding p-value. If p was less than 0.05, we considered the attributes compared to be statistically significant in impacting either the final game score or the number of turns to achieve the maximum defender score.

1. Effect of Attacker Profile

The average total scores of APTs showed that some were easier to defend against based on their resource and stealth levels, defined in IV.A.1 (Table 11). Attacker resource levels have a statistically significant effect on the final total game scores (Table 12). This reflects the differences in sponsorship associated with these actors, as those that are state or state-sponsored typically performed better than non-state. We also observed that the attacker's concern for stealth had a statistically significant effect on the final total game scores (Table 13). However, APT X is an exception. Although it has a high resource value assignment, it also has a high concern-for-stealth value. This imposed higher cost when the attacker took actions designated as less stealthy, increasing the defending player's final score. This is replicated across other results, with players assigned a higher concern-for-stealth value incurring higher costs, which resulted in lower final scores.

Table 11. Average total scores for groups of APTs based on APT attributes. APT test groups are the abbreviated APTs (described in Table 5). APT group attributes are those defined in IV.A.1. Average final scores are extracted from Table 7.

APT Test Group	APT Group Attribute	Average Final Score
RK, ST, UC	Resource level = 3	72.51
LG, MW, Tur, Zirc	Resource level = 2	154.68
WS	Resource level = 1	113.53
MW, Zirc, UC	Stealth level = 3	227.44
LG, Tur, WS	Stealth level = 2	88.27
RK, ST	Stealth level = 1	1.33

Table 12. Statistical significance of the APT resources attribute (described in Chapter IV.A.1). APT abbreviations are from Table 5. P-values were calculated from a t-test on average final game scores (from Table 7) of compared groups.

APT Groups Compared by Resources Attribute	P-value	Avg Final Scores
Low resources: WS Med resources: LG, MW, TUR, ZIRC	7.83E-03	Low: 113.53 Med: 154.68
Med resources: LG, MW, TUR, ZIRC High resources: RK, SWT, UC	3.12E-02	Med 154.68 High: 72.51
High resources: RK, SWT, UC Low resources: WS	7.15E-04	High: 72.51 Low: 113.53

Table 13. Statistical significance of APT concern for stealth attribute (described in Chapter IV.A.1). APT abbreviations are from Table 5. P-values were calculated from a T-test on average final game scores (from Table 7) of compared groups.

APT Groups Compared by Stealth Attribute	P-value	Avg Final Scores
Low stealth: RK, SWT Med stealth: LG, TUR, WS	2.39E-07	Low: 1.33 Med: 88.27
Med stealth: LG, TUR, WS High stealth: MW, ZIRC, UC	7.80E-15	Med: 88.27 High: 227.44
High stealth: MW, ZIRC, UC Low stealth: RK, SWT	4.07E-08	High: 227.44 Low: 1.33

2. Effect of Game Length

Across all 5,304 games, those with turn limits greater than 200 turns offered the best opportunities for the defender to maximize their score against the attacker. Therefore, lengthening the game helps the defender but this is true only up to a point. As to real time, each action during a turn has varying associated duration (e.g. a security policy can be implemented quickly but a virtualized network requires significant setup time).

Despite lacking a direct conversion to real time, lengthening the game's turns helps defend against APTs since APTs tend to persist even when they know they are engaging with an active defender. The benefit has risks though, as the score difference between attacker and defender becomes wider as the games progress; while the defender may perform well in these games, this added length affords attackers the opportunity to perform their most exquisite actions in later phases, reflecting accomplishment of their ultimate objective. In general, we saw that an attacker with sufficient turns and time will eventually outperform the defender past a certain inflection point.

Defender attributes (Table 14) showed inconsistent trends. Defenders with high resource attribute values did not take fewer turns to maximize their score. The

confidentiality-integrity-availability ordering had no statistically significant impact on average number of turns to maximize defender score.

Unexpectedly, defender security was not seen to significantly affect when the defender maximized their scores (Table 15), or average total score (Table 16). This could be because our initial states do not realistically reflect security levels, or because they need additional attributes. Both players may have been unable to use some actions because of a missing initial state condition in some games. A way to improve this could be to use the previous game state of an interaction as a start of the next one, to simulate gained experience.

Table 14. Average number of turns for defender groups band some attribute values. Group attribute values (resources, preferences, and initial security state) are described in Chapter IV.A.1. Average number of turns indicates the average turn number at which the defenders had the highest final game score (from Table 8).

Defender Attribute Groups	Group Attribute Value	Average Number of Turns
A, D, G, J, K, L, M	Resource level = low	247
B, E, H	Resource level = medium	255
C, F, I	Resource level = high	250
A, B, C, J, K	(C, I, A) preference	254
D, E, F, L	(I, C, A) preference	254
G, H, I, M	(A, I, C) preference	240
A, B, C	Initial security state = high	248
D, E, F	Initial security state = medium	253
G, H, I	Initial security state = low	238
J, K, L, M	Initial security state = very low	258

Table 15. Statistical significance of average max turns for optimized defender performance, based on differing defender attributes (described in IV.A.1). Defenders were grouped by attributes from Table 6. Average max turns (from Table 8) were used to derive p-values.

Defender Test Grouping	Avg Max Turn P-value	Avg Max Turn
Low resources: A, D, G, J, K, L, M Medium resources: B, E, H	0.31	Low: 247.57 Med: 254.67
Med resources: B, E, H High resources: C, F, I	0.40	Med: 254.67 High: 250.33
High resources: C, F, I Low resources: A, D, G, J, K, L, M	0.42	High: 250.33 Low: 247.57
(C, I, A) preference: A, B, C, J, K (I, C, A) preference: D, E, F, L	0.44	(C, I, A): 253.5 (I, C, A): 255
(I, C, A) preference: D, E, F, L (A, I, C) preference: G, H, I, M	0.16	(I, C, A): 255 (A, I, C): 240.25
(C, I, A) preference: A, B, C, J, K (A, I, C) preference: G, H, I, M	0.16	(C, I, A): 253.5 (A, I, C): 240.25
High security: A, B, C Med security: D, E, F	0.38	High: 248 Med: 253
Med security: D, E, F Low security: G, H, I	0.15	Med: 253 Low: 238
Low security: G, H, I Very low security: J, K, L, M	0.13	Low: 238 Very low: 257.75
High security: A, B, C Very low security: J, K, L, M	0.15	High: 248 Very low: 257.75

Table 16. Statistical significance of average total score for optimized defender performance, based on differing defender attributes (described in Chapter IV.A.1). Defenders were grouped by attributes from Table 6. Average final game scores (from Table 7) were used to derive p-values.

Defender Test Grouping	Avg Total Score P-Value	Avg Final Score
Low resources: A, D, G, J, K, L, M Medium resources: B, E, H	0.30	Low: 87.44 Med: 143.01
Med resources: B, E, H High resources: C, F, I	0.28	Med: 143.01 High: 167.43
High resources: C, F, I Low resources: A, D, G, J, K, L, M	0.27	High: 167.43 Low: 87.44
(C, I, A) preference: A, B, C, J, K (I, C, A) preference: D, E, F, L	0.43	(C, I, A): 141.92 (I, C, A): 121.29
(I, C, A) preference: D, E, F, L (A, I, C) preference: G, H, I, M	0.33	(I, C, A): 121.29 (A, I, C): 87.16
(C, I, A) preference: A, B, C, J, K (A, I, C) preference: G, H, I, M	0.47	(C, I, A): 141.92 (A, I, C): 87.16
High security: A, B, C Med security: D, E, F	0.03	High: 140.71 Med: 128.14
Med security: D, E, F Low security: G, H, I	0.05	Med: 128.14 Low: 101.32
Low security: G, H, I Very low security: J, K, L, M	0.31	Low: 101.32 Very low: 108.23
High security: A, B, C Very low security: J, K, L, M	0.21	High: 140.71 Very low: 108.23

3. Recommended Defensive Techniques

The defender had 23 deception, 7 detection, 41 hardening, and 7 moving-target actions available in the game. The top 10 highest-scoring defensive actions (shown in Table 9) included 9 deception actions, 1 hardening action, no moving-target defenses, and no detection actions (Table 17). Deception actions scored well for all defenders and circumstances.

Across the APT life-cycle phases, the highest-scoring defensive techniques varied by phase (Table 18). In our tests, it was most effective for the defender to deceive early in the APT's lifecycle, particularly during reconnaissance. This suggests that active cyber defense is less effective later in the lifecycle, when an APT is already inside a target network. Phases 3 and 4 rely on hardening techniques, suggesting cyber deception is less

effective once an APT has established a foothold in a target network. These results are consistent with the defense-in-depth strategy (Mughal, 2018).

Table 17. Action class membership percentages for top 10 highest-scoring actions set versus total actions set. Percentages are derived from Table 9.

Action Class	% in top actions	% of total actions
Cyber deception	90%	29.5%
Hardening	10%	52.5%
Detection	0%	9%
Moving-target defense	0%	9%

Table 18. Action class membership percentages for highest-scoring defensive techniques by APT life-cycle phase. Percentages are derived from Table 10.

Phase	Cyber Deception	Hardening	Detection	Moving-target defense
Reconnaissance	100%	0%	0%	0%
Establishing-a-foothold	40%	20%	20%	20%
Lateral movement	20%	80%	0%	0%
Data exfiltration	0%	100%	0%	0%

4. Effect of Defender Resource Level

When comparing defenders of varying resource levels without varying other parameters, defenders with higher assigned values for resources scored appeared to score better against APT attackers (Figure 14). This suggests that increased spending on cybersecurity can reduce cyber risks (Asen et al., 2019). However, using the t-test we found

that defender's resource value assignment was not statistically significant in impacting the final score, indicating that other factors are more important (Table 16).

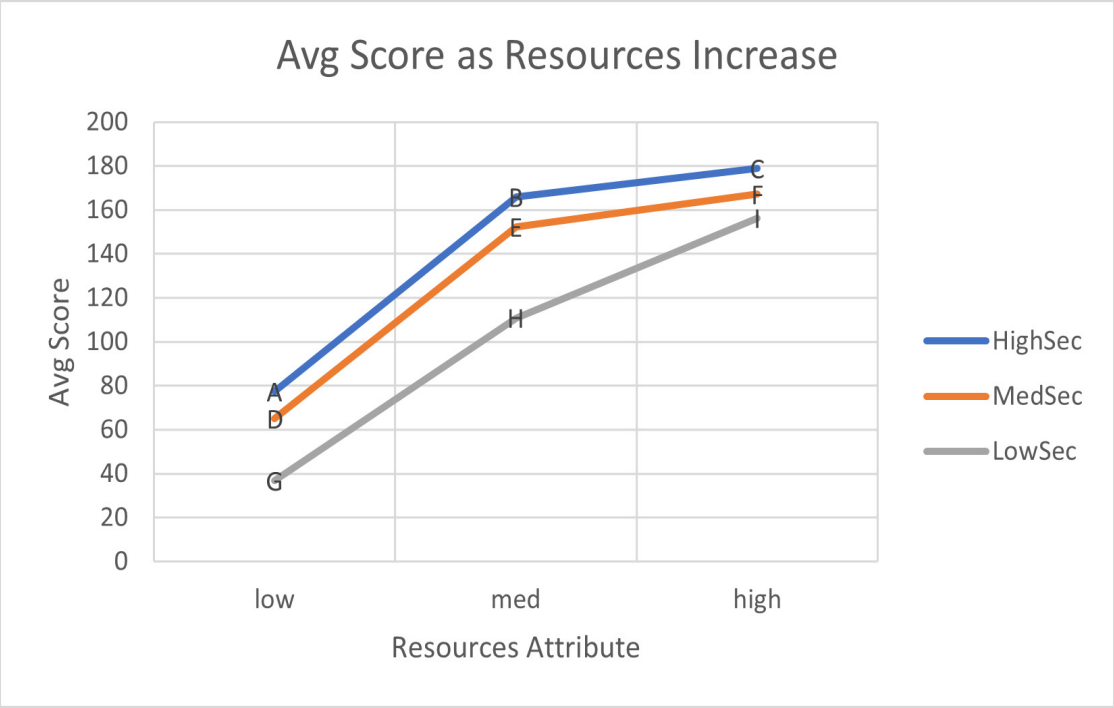

Figure 14. Effect of changes to defender resource attribute (described in Chapter IV.A.1). Defenders were grouped by attributes from Table 6. Each line shows a change in the resources attribute only. Initial security state and confidentiality, integrity, and availability priorities were constant. Lines are labeled with abbreviations for defenders A-I. Average scores are extracted from Table 7.

5. Effect of Defender (C, I, A) Priority

When comparing defenders of differing priorities of confidentiality, integrity, and availability, scores were high when confidentiality or integrity was the top priority (Table 19). This supports the well-known result that a high emphasis on keeping systems available often conflicts with security since systems must be available to be attacked. This reflects a common dilemma in cybersecurity as priorities on confidentiality, integrity, and availability can conflict with those of business operations. The t-test indicated that the value assignment for confidentiality-integrity-availability did not have a statistically

significant impact on game outcomes, and that another factor was animating the varied results (Table 15 and Table 16).

Table 19. Average final scores by confidentiality, integrity, and availability priority groups (described in Chapter IV.A.1). Defenders were grouped by attributes from Table 6. Average scores are extracted from Table 7.

Defender Grouping	(C, I, A) Priority Ordering	Average Final Score
A, B, C, J, K	(C, I, A)	141.92
D, E, F, L	(I, C, A)	121.29
G, H, I, M	(A, I, C)	87.16

6. Effect of Defender's Initial Security State

Defenders of differing security states at the start showed noticeable differences in final score (Table 20). As expected, the highest initial defender security states typically produced the best defender score. An exception was the very-low security state of no defensive preparations. This could be because the defender's costs are significantly lower, or the attacker has more options of lower benefits then. Statistically significant impact on game scores was observed between high and medium, medium and low initial security states. Differences between very low initial security states and low security states did not show statistically significant impact on game outcomes. This suggests that the best way to defend against an attacker is to ensure the initial security state of the system includes as few vulnerabilities as possible, and as many security controls as are feasible.

Table 20. Final score averages for abbreviated defenders grouped by initial security state level (described in Chapter IV.A.1). Defenders were grouped by attributes from Table 6. Average scores are extracted from Table 7.

Defender Grouping	Initial Security State Level	Average Final Score
A, B, C	High	171.91
D, E, F	Medium	126.47
G, H, I	Low	101.32
J, K, L, M	Very Low	108.16

VI. CONCLUSION

We showed that game theory with reinforcement learning can model active cyber defenses against APTs. Of the active techniques, cyber deception was the most useful for defenders, mostly in the early phases of an APT campaign; hardening techniques were better in the later phases. Defender attributes of resource level and confidentiality-integrity-availability priority had less impact on defender performance than the defender's initial security state, APT resource level, or concern for stealth. The emergence of a defender maximum score in many games shows that it is often valuable for a defender to lengthen games. But with enough time, the attacker will continually improve.

Our game model is an extensible planning tool for cyber defenders. It can evaluate many defensive strategies against many APTs with different origins, goals, and preference profiles. It is compatible with other phased models of APT attacks such as the cyber kill chain. While we primarily drew actions from MITRE ATT&CK and MITRE D3FEND frameworks, adding action options from other sources could improve the model. Action profiles can also be adjusted to reflect new costs and benefits to an attacker or defender. Duration of actions could be measured from observations of real attacks and defensive preparations. This would help determine on average how long a defender should continue to interact with an APT to learn from it.

Future work should also use data from real-time network traffic where it is unknown whether an APT is present, using techniques appropriate for the anticipated phase, and observing changes to confirm suspicions. This could include classifying traffic by unsupervised learning such as reinforcement learning (Shen et al., 2023).

Our model could be extended to permit active defender actions to invalidate attacker accomplishments and return the attacker to previous phases of the APT attack lifecycle. It could also include techniques to counter open-source intelligence or physical-security attack vectors. Future work could also include the industrial-control-system and mobile-device tactics that MITRE has identified.

To make our model more realistic, future work could also include delays on both sides. For example, an APT may stay idle in a system after establishing a foothold, without elevating privileges or moving laterally for a long time. Our model could also allow attackers and defenders to have multiple simultaneous activities. Our model could also handle cases where the attacker and defender have partial knowledge of the other's strategies; for example, it would help to understand how an APT would respond if it knew that the defender was deceiving. Future work could use conditional probabilities to make decisions for the players. Two types of possible deception failures from the defender's perspective are: failure of the attacker to notice the deception and taking the notice as a challenge. Incorporation of conditional probabilities could capture these by modeling how attackers respond to being deceived. Angela Tan at our school is currently investigating this topic.

APPENDIX A: INPUT DATA

A. PLAYER PROFILES

The following attacker player specifications were generated and tested.

- ./cases/apt39/attacker_specs.csv
- ./cases/lazarus/attacker_specs.csv
- ./cases/muddywater/attacker_specs.csv
- ./cases/sandworm/attacker_specs.csv
- ./cases/turla/attacker_specs.csv
- ./cases/wizard/attacker_specs.csv
- ./cases/zirconium/attacker_specs.csv
- ./cases/use_case/attacker_specs.csv

The following attacker player specifications were generated and tested.

- ./defenders/Alpha_defender_specs.csv
- ./defenders/Bravo_defender_specs.csv
- ./defenders/Charlie_defender_specs.csv
- ./defenders/Delta_defender_specs.csv
- ./defenders/Echo_defender_specs.csv
- ./defenders/Foxtrot_defender_specs.csv
- ./defenders/Golf_defender_specs.csv
- ./defenders/Hotel_defender_specs.csv

- ./defenders/India_defender_specs.csv
- ./defenders/Juliet_defender_specs.csv
- ./defenders/Kilo_defender_specs.csv
- ./defenders/Lima_defender_specs.csv
- ./defenders/Mike_defender_specs.csv

B. INITIAL SECURITY STATES

The following are the initial security states used for our game model.

initial_state_high.csv	**initial_state_medium.csv**	**initial_state_low.csv**	**initial_state_very low.csv**
used_dig_sig	got_suscept_hum_targ	found_vuln_website	found_vuln_website
used_mac	obtained_net_info	got_suscept_hum_targ	got_suscept_hum_targ
used_crypto_system	got_contact_info	obtained_net_info	obtained_net_info
used_auth_server	got_net_access	got_physcl_access	got_physcl_access
used_av_software	reached_net	got_contact_info	got_contact_info
used_manage_serv	used_dig_sig	obtained_credentials	obtained_credentials
implem_firewall	used_mac	found_vuln_software	found_vuln_software
implem_hw_iso	used_crypto_system	got_net_access	got_net_access
implem_io_restrict	used_auth_server	found_vuln_net_dvc	found_vuln_net_dvc
implem_kern_iso	used_av_software	reached_net	reached_net
implem_bcast_iso	implem_firewall	implem_firewall	
implem_dns_filter	implem_io_restrict	used_mac	
used_backups	implem_bcast_iso	used_auth_server	
implem_proc_seg	implem_dns_filter	used_av_software	
used_otp_infrastruct	used_backups	implem_bcast_iso	
used_bio_auth_dvc	used_otp_infrastruct	established_domains	
established_domains	established_domains		
created_hash_whitelist	created_hash_whitelist		
created_authent_baseline			
created_authoriz_baseline			

APPENDIX B: CODE

DataPreprocessor(dataset, scalar). This module takes a dataset name ('DAPT2020' or 'SCVIC') and a sklearn scalar object (StandardScalar(), RobustScalar(), etc.) and produces a Pandas dataframe that is preprocessed and scaled for use in classification.

- dataset: string describing the dataset name ('DAPT2020' or 'SCVIC').
- scalar: a sklearn scalar object.

APTClassifier(algorithm, tts). This module trains and validates a classifier of the provided algorithm, producing a classification report, confusion matrix, and accuracy score. A function to produce a data manifold visualization using t-SNE is also provided.

- algorithm: a string that is the three-letter abbreviation of the desired classifier.
- tts: a number less than 1 and greater than 0, that describes the training-test set split.

Attacker(name, resources, stealth, preferred_tactics, pref_weight). This data structure holds information about an attacker player in the game model.

- name: a string providing the attacker player's name.
- resources: an integer as described in IV.A.1.
- stealth: an integer as described in IV.A.1.
- preferred_tactics: a set of strings as described in IV.A.1.
- pref_weight: a floating-point number, as described in IV.A.1.

Defender(name, resources, cia). This data structure holds information about a defender player in the game model.

- name: a string providing the defender player's name.

- resources: an integer as described in IV.A.1.
- cia: a string describing the defender's priority list, corresponding to confidentiality, integrity, and availability preferences.

CBA(attacker_obj, defender_obj). This module provides functions that turn attacker objects into costs and benefits based on their attributes. At least one parameter must be provided.

- attacker_obj: an initialized attacker object.
- defender_obj: an initialized defender object.

Defender_builder(defender_obj). This module outputs a defender_spec.csv based on the defender object provided, using the CBA module's functions to produce costs and benefits.

- defender_obj: an initialized defender object.

Mitre_spec(). This module produces a library of attacker objects, imports their preferred actions from a MITRE enterprise layer, and outputs attacker_specs.csv for each provided .json layer.

The attack_visualizer.py file builds dot language based .png or .svg directed graph visualizations of all attackers in the cases directory created by mitre_spec. Both are placed in the viz/ folder inside the cases/[attacker_name]/ folder. The SVGs are most useful as the tooltip shows the linkages.

A. GAME MODELING

Modified code from that written by Professor Rowe used to perform gameplay and reinforcement learning is in decepgame.py. We modified decepgame(attacker_spec, attacker_name, defender_spec, defender_name, initial_state, numgames, maxmoves, timenow, sd):

- attacker_spec: the path to the attacker specification CSV.

- attacker_name: string of the attacker's name.
- defender_spec: the path to the defender specification CSV.
- initial_state: the path the initial state CSV.
- numgames: the number of games to play before reporting a score.
- maxmoves: the maximum number of moves to play per game.
- timenow: a string providing the start time for results reporting.

B. INSTRUMENTATION AND RESULTS PROCESSING

The run_tests.py program tests using the testing_schedule.csv and records results to a timestamped CSV, as well as generates figures that plot final comparative benefits per interaction (series of games) and final scores per interaction with a trendline. Tests are run as new processes that write to the final results file. The multiprocessing version (run_tests_mp.py) requires the ability to handle as many simultaneous processes as tests (our test schedule included 104, and we used the NPS Hamming high-performance computer to run them). Tests are listed in testing_schedule.csv, and the run_test_mp.py program loads the testing schedule, builds the required specifications for attackers and defenders, and executes the decepgame program to run the tests, recording results of each run of the decepgame function in /results/[date/time of the test]/test_results_[date/time of the test].csv. Each attacker-defender pair has a folder containing plots of attacker and defender benefits, a scatter plot of game scores in .png and .svg formats. output_control.py provides a few useful functions for controlling the output to standard output, as well as controlling the output to files.

The files parse_final.py, and build_tables.py support results processing and formatting.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Achleitner, S., La Porta, T., McDaniel, P., Sugrim, S., Krishnamurthy, S., & Chadha, R. (2017). Deceiving network reconnaissance using SDN-based virtual topologies. *IEEE Transactions on Network and Service Management*, *14*(4), 1098–1112. https://doi.org/10.1109/tnsm.2017.2724239

Alshamrani, A., Myneni, S., Chowdhary, A., & Huang, D. (2019). A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. *IEEE Communications Surveys and Tutorials*, *21*(2), 1851–1877. https://doi.org/10.1109/COMST.2019.2891891

Aparicio-Navarro, F. J., Kyriakopoulos, K. G., Ghafir, I., Lambotharan, S., & Chambers, J. A. (2018, October 1). Multi-stage attack detection using contextual information. *IEEE Xplore.* https://doi.org/10.1109/MILCOM.2018.8599708

AppOmni, Netskope, Praetorian. (2022, October 18). *Data from cloud storage*. MITRE. http://attack.mitre.org/techniques/T1530/

Amin, A., Shetty, S., Njilla, L., Tosh, D., & Kamhoua, C. (2021). Hidden Markov model and cyber deception for the prevention of adversarial lateral movement. *IEEE Access, 9*, 49662–49682. https://doi.org/10.1109/access.2021.3069105

Anagnostakis, K., Sidiroglou, S., Akritidis, P., Xinidis, K., Markatos, E., & Keromytis, A. (2005). Detecting targeted attacks using shadow honeypots. *USENIX Security Symposium* (p. 9). https://doi.org/10.7916/d8wm1ps8

Asen, A., Bohmayr, W., Deutscher, S., González, M., & Mkrtchian, D. (2019). *Are you spending enough on cybersecurity?* https://www.bcg.com/publications/2019/are-you-spending-enough-cybersecurity.

Aven, T. (2013). On the meaning of a black swan in a risk context. *Safety Science*, *57*, 44–51.

Avertium. (2022, April 13). *An in-depth look at Iranian APT "MuddyWater."* https://explore.avertium.com/resource/in-depth-look-at-iranian-apt-muddywater

Bai, T., Bian, H., Daya, A., Salahuddin, M., Limam, N., & Boutaba, R. (2019). A machine learning approach for RDP-based lateral movement detection. *IEEE 44th Conference on Local Computer Networks* (pp. 242–245). doi: 10.1109/LCN44214.2019.8990853.

Bao L. (2016). Boosted near-miss under-sampling on SVM ensembles for concept detection in large-scale imbalanced datasets. *Neurocomputing*, 198–206. https://doi.org/10.1016/j.neucom.2014.05.096

Bazrafshan, Z., Hashemi, H., Fard, S, & Hamzeh, A. (2013). A survey on heuristic malware detection techniques. *The 5th Conference on Information and Knowledge Technology* (pp. 113–120). doi: 10.1109/IKT.2013.6620049.

Bell, E. (2019). Malicious digital penetration of United States weaponized military unmanned aerial vehicle systems: A national security perspective concerning the complexity of military UAVs and hacking. *Mathematics and Computer Science Capstones,* 46.

Blagden, D. (2020). Deterring cyber coercion: The exaggerated problem of attribution. *Survival, 62*(1), 131–148. doi:10.1080/00396338.2020.1715072

Bhuyan, M., Bhattacharyya, D., & Kalita, J. (2011). Surveying port scans and their detection methodologies. *The Computer Journal, 54*(10), 1565–1581. https://doi.org/10.1093/comjnl/bxr035

Bi, J., He, S., Luo, F., Meng, W., Ji, L., & Huang, D. (2022). Defense of advanced persistent threat on industrial Internet of Things with lateral movement modelling. *IEEE Transactions on Industrial Informatics,* 1–11. https://doi.org/10.1109/tii.2022.3231406

Bourke, D., & Grzelak, D. (2018). *Breach detection at scale with AWS honey tokens*. Blackhat Asia, https://www. blackhat.com/asia-18/briefings/htmlbreach-detection-at-scale-withaws-honey-tokens, 20–23.

Birkinshaw, C., Rouka, E., & Vassilakis, V. (2019). Implementing an intrusion detection and prevention system using software-defined networking: Defending against port-scanning and denial-of-service attacks. *Journal of Network and Computer Applications, 136*, 71–85. https://doi.org/10.1016/j.jnca.2019.03.005

Brogi, G., & Bernardino, E. D. (2019). Hidden Markov models for advanced persistent threats. *International Journal of Security and Networks*, *14*(4), 181–190.

Brogi, G., & Tong, V. (2016). TerminAPTor: Highlighting advanced persistent threats through information flow tracking. *2016 8th IFIP International Conference on New Technologies, Mobility and Security* (pp. 1-5). doi: 10.1109/NTMS.2016.7792480.

Caltagirone, S., Pendergast, A., & Betz, C. (2013). *The diamond model of intrusion analysis*. Center For Cyber Intelligence Analysis and Threat Research.

Chadza, T., Kyriakopoulos, K., & Lambotharan, S. (2020). Analysis of hidden Markov model learning algorithms for the detection and prediction of multi-stage network attacks. *Future Generation Computer Systems*, *108*, 636–649. https://doi.org/10.1016/j.future.2020.03.014

Cho, J., Sharma, D., Alavizadeh, H., Yoon, S., Ben-Asher, N., Moore, T., Kim, D., Lim, H., & Nelson, F. (2020). Toward proactive, adaptive defense: A survey on moving target defense. *IEEE Communications Surveys & Tutorials, 22*(1), 709–745. https://doi.org/10.1109/comst.2019.2963791

Chu, W., Lin, C., & Chang, K. (2019). Detection and classification of advanced persistent threats and attacks using the support vector machine. *Applied Sciences, 9*(21), *4579.* https://doi.org/10.3390/app9214579

Cojocar, L., Loughlin, K., Saroiu, S., Kasikci, B., & Wolman, A. (2021). MFIT: A bump-in-the-wire tool for plug-and-play analysis of Rowhammer susceptibility factors. *Technical Report–Microsoft Research.*

Crouse, M., Prosser, B., & Fulp, E. W. (2015). Probabilistic performance analysis of moving target and deception reconnaissance defenses. *In Proceedings of the Second ACM Workshop on Moving Target Defense.* https://doi.org/10.1145/2808475.2808480

Cunningham, C. (2020). *A Russian Federation information warfare primer*. Henry M. Jackson School of International Studies. Washington University.

Cyber Security and Infrastructure Agency. (2023). *CISA releases Decider Tool to help with MITRE ATT&CK mapping*. https://www.cisa.gov/news-events/alerts/2023/03/01/cisa-releases-decider-tool-help-mitre-attck-mapping

Dargahi, T., Dehghantanha, A., Bahrami, P., Conti, M., Bianchi, G., & Benedetto, L. (2019). A cyber-kill-chain based taxonomy of crypto-ransomware features. *Journal of Computer Virology and Hacking Techniques, 15*(4), 277–305. https://doi.org/10.1007/s11416-019-00338-7

Dietrich, C., Rossow, C., Freiling, F., Bos, H., Van Steen, M., & Pohlmann, N. (2011). On botnets that use DNS for command and control. *2011 Seventh European Conference on Computer Network Defense.* https://doi.org/10.1109/ec2nd.2011.16

DiMaggio, J. (2021). *Ransom mafia. Analysis of the world's first ransomware cartel*. Analyst white paper.

Doherty, N., Anastasakis, L., & Fulford, H. (2009). The information security policy unpacked: A critical study of the content of university policies. *International Journal of Information Management, 29*(6), 449–457. https://doi.org/10.1016/j.ijinfomgt.2009.05.003

Drew, S., & Heinen, C. (2022). *Testing deception with a commercial tool simulating cyberspace*. http://hdl.handle.net/10945/69636

Eckhardt, R. & Bagui, S. *AI, machine learning and deep learning: A security perspective.* University of West Florida. doi: 10.1201/9781003187158-23

Faou, M. (2019). *Turla lightneuron: One email away from remote code execution*. ESET Research white paper.

Fearon, J. (1995). Rationalist explanations for war. *International Organization, 49*(3), 379–414. http://www.jstor.org/stable/2706903

Fei, Y., Ning, J., & Jiang, W. (2018). A quantifiable attack-defense trees model for APT attack. *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference* (pp. 2303–2306). doi: 10.1109/IAEAC.2018.8577817.

Fenrich, K. (2008). Securing your control system: the "CIA triad" is a widely used benchmark for evaluating information system security effectiveness. *Power Engineering,* 112(2), *44*+. https://link.gale.com/apps/doc/A177028777/AONE?u=anon~e527e3a2&sid=googleScholar&xid=8cc52b63

Ferguson-Walter, K., Fugate, S., Mauger, J., & Major, M. (2019). Game theory for adaptive defensive cyber deception. *Proceedings of the 6th Annual Symposium on Hot Topics in the Science of Security.* doi: 10.1145/3314058.

Ferguson-Walter, K., Major, M., Johnson, C., & Muhleman, D. (2021, August). Examining the efficacy of decoy-based and psychological cyber deception. *In USENIX Security Symposium* (pp. 1127–1144).

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). *Learning from Imbalanced Data Sets*. Springer.

Fischerkeller, M., & Harknett, R. (2017). Deterrence is not a credible strategy for cyberspace. *Orbis*, *61*(3), 381–393.

Fonseca, F., Pinto, R., & Meira, W. (2005, October). Increasing user's privacy control through flexible web bug detection. *In Third Latin American Web Congress* (pp. 8-pp). IEEE.

Giura, P., & Wang, W. (2012). A context-based detection framework for advanced persistent threats. *2012 International Conference on Cyber Security* (pp. 69–74). doi: 10.1109/CyberSecurity.2012.16.

Green, Justin J. (2020). *The fifth masquerade: An integration experiment of military deception theory and the emergent cyber domain*. http://hdl.handle.net/10945/66078

Haig, L. (2022). *LaBrea: A new approach to securing our networks*. SANS Institute white paper.

Hare, F., & Diehl, W. (2020). *Noisy operations on the silent battlefield: Preparing for adversary use of unintrusive precision cyber weapons*. *The Cyber Defense Review,* 5(1), 153–168. https://www.jstor.org/stable/26902668

Hassan, S., & Guha, R. (2016). Modelling of the state of systems with defensive deception. *2016 International Conference on Computational Science and Computational Intelligence.* doi: 10.1109/csci.2016.0197.

Hasham, S., Joshi, S., & Mikkelsen, D. (2019). *Financial crime and fraud in the age of cybersecurity*. McKinsey & Company.

Hawley, S., Read, B., Brafman-Kittner, C., Fraser, N., Thompson, A., Rozhansky, Y., & Yashar, S. (2019, January 29). *APT39: An Iranian cyber espionage group focused on personal information*. https://www.mandiant.com/resources/blog/apt39-iranian-cyber-espionage-group-focused-on-personal-information

Heckman, K., Stech, F., Thomas, R., Schmoker, B., & Tsow, A. (2015). *Cyber denial, deception and counter deception* (1st ed.). doi:10.1007/978-3-319-25133-2

Heitzenrater, C., Taylor, G., & Simpson, A. (2015). When the winning move is not to play: Games of deterrence in cyber security. *Lecture Notes in Computer Science,* 250–269. doi: 10.1007/978-3-319-25594-1_14.

Hinkle, K. (2011). Countermeasures in the cyber context: One more thing to worry about. *Yale Journal of International Law, 37* (Fall), 11–21.

Hsu, H. and Lachenbruch, P.A. (2014). Paired t test. Wiley StatsRef: Statistics Reference Online. https://doi.org/10.1002/9781118445112.stat05929

Hu, P., Li, H., Fu, H., Cansever, D., & Mohapatra, P. (2015). Dynamic defense strategy against advanced persistent threat with insiders. *2015 IEEE Conference on Computer Communications.* doi: 10.1109/infocom.2015.7218444

Hutchins, E., Cloppert, M., & Amin, R. (2011). Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research, 1*(1), 80.

Hyder, M., & Ismail, M. (2021). Securing control and data planes from reconnaissance attacks using distributed shadow controllers, reactive and proactive approaches. *IEEE Access,* 9, *21881–21894.* https://doi.org/10.1109/access.2021.3055577

Ibrahima, I., Mandiant, Praetorian, Elwell, R. (2022). *Cloud infrastructure discovery*. MITRE. https://attack.mitre.org/techniques/T1580/

Joint Chiefs of Staff. (2018). CJCSM 6510.01B: *Cyber incident handling program*. https://public.cyber.mil/policy-guidance/cjcsm_6510-01b/

Joint Chiefs of Staff. (2018). Joint Publication 3-12: *Cyberspace operations*. Department of Defense.

Joint Task Force. (2021). NIST SP 800-53 revision 5: *Security and privacy controls for federal information systems and organizations*. National Institute of Standards and Technology, Special Publication 800-53 Revision 5.

Joint Task Force Transformation Initiative. (2011). NIST SP 800-39: *Managing information security risk. Organization, Mission, and Information System View, National Institute of Standards and Technology*. https://csrc.nist.gov/publications/detail/sp/800-39/final

Kahn, R.., McConnell, M., Nye, J., Schwartz, P., Daly, N., Fick, N., Finnemore, M., Fontaine, R., Geer, D., Gross, D., Healey, J., Lewis, J., Lucarelli, M., Mahnken, T., McGraw, G., Miksad, R., Rattray, G., Rogers, W., & Schroeder, C. (2011). U.S. national interests in cyberspace. *K. M. Lord & T. Sharp (Eds.), America's Cyber Future: Security and Prosperity in the Information Age* (pp. 12–19). Center for a New American Security. http://www.jstor.org/stable/resrep06319.6

Kaloroumakis, P., & Smith, M. (2021). *Toward a knowledge graph of cybersecurity countermeasures*. The MITRE Corporation.

Kennedy, D., O'Gorman, J., Kearns, D., & Aharoni, M. (2011). *Metasploit: the penetration tester's guide*. No Starch Press.

Kim, Y., Lee, J., Go, M., & Lee, K. (2020, October). Analysis of the asymmetrical relationships between state actors and apt threat groups. *In 2020 International Conference on Information and Communication Technology Convergence* (pp. 695–700). IEEE.

Kumar, R., Singh, S., & Kela, R. (2021). A quantitative security risk analysis framework for modelling and analyzing advanced persistent threats. *In Foundations and Practice of Security: 13th International Symposium, FPS 2020, Canada, December 1–3, 2020, Revised Selected Papers 13* (pp. 29–46). Springer International Publishing.

Lackner, P. (2021). How to mock a bear: Honeypot, honeynet, honeywall & honeytoken: A survey. *International Conference on Enterprise Information Systems.* https://doi.org/10.5220/0010400001810188

Landsborough, J., Carpenter, L., Coronado, B., Fugate, S., Ferguson-Walter, K., & Van Bruggen, D. (2021). *Towards self-adaptive cyber deception for defense*. University of Hawaii. https://hdl.handle.net/10125/70857

Lehto, M. (2022). APT cyber-attack modelling: Building a general model. *Proceedings of the International Conference on Information Warfare and Security,* 17(1), *121–129.* https://doi.org/10.34190/iccws.17.1.36

Lei, C., Zhang, H.-Q., Tan, J., Zhang, Y., & Liu, X. (2018). Moving target defense techniques: A survey. *Security and Communication Networks 2018,* 1–25. https://doi.org/10.1155/2018/3759626

Li, J., Malcom Vetter, T., Lesicki, W. (2022, October 18). *Account manipulation*. MITRE. https://attack.mitre.org/techniques/T1098/

Li, P., & Yang, X. (2019). On dynamic recovery of cloud storage system under advanced persistent threats. *IEEE Access,* 7, *103556-103569.*

Lin, J., Chen, J., Chen, C., & Chien, Y. (2009). A game theoretic approach to decision and analysis in strategies of attack and defense. *2009 Third IEEE International Conference on Secure Software Integration and Reliability Improvement, China* (pp. 75–81). doi: 10.1109/SSIRI.2009.27.

Liu, J., Shen, Y., Simsek, M., Kantarci, B., Mouftah, H., Bagheri, M., & Djukic, P. (2022). A new realistic benchmark for advanced persistent threats in network traffic. *IEEE Networking Letters, 4*(3), 162–166. https://doi.org/10.1109/lnet.2022.3185553

Lynn, W. (2010). Defending a new domain: The Pentagon's cyberstrategy. *Foreign Affairs,* 89(5), 97–108. http://www.jstor.org/stable/20788647

MacFarland, D., & Shue, C. (2015). The SDN shuffle. *Proceedings of the Second ACM Workshop on Moving Target Defense.* https://doi.org/10.1145/2808475.2808485

Mandiant. (2021). *Exposing one of China's cyber espionage units.* https://www.mandiant.com/sites/default/files/2021-09/mandiant-apt1-report.pdf

Mandiant. (2022a). *Advanced persistent threats: Threat actors and groups*. Mandiant. https://www.mandiant.com/resources/insights/apt-groups

Mandiant. (2022b). Unc2452 merged into apt29. Mandiant. https://www.mandiant.com/resources/blog/unc2452-merged-into-apt29

Marchetti, M., Pierazzi, F., Colajanni, M., & Guido, A. (2016). Analysis of high volumes of network traffic for advanced persistent threat detection. *Computer Networks*, *109*, 127–141. https://doi.org/10.1016/j.comnet.2016.05.018

McKenzie, T. (2017). *Front matter: Is cyber deterrence possible?* (pp. i–ii). Air University Press. http://www.jstor.org/stable/resrep13817.1

Meicong, L., Huang, W., Wang, Y., Fan, W., & Li, J. (2016). The study of APT attack stage model. *Annual ACIS International Conference on Computer and Information Science.* https://doi.org/10.1109/icis.2016.7550947

Meier, J., Nguyen, T., & Rowe, N. (2023). Hardening honeypots for industrial control systems. *Hawaii International Conference on Systems Sciences.*

Mell, P., Scarfone, K., & Romanosky, S. (2006). Common vulnerability scoring system. *IEEE Security & Privacy, 4*(6), 85–89. doi: 10.1109/MSP.2006.145

Min, M., Xiao, L., Xie, C., Hajimirsadeghi, M., & Mandayam, N. B. (2018). Defense against advanced persistent threats in dynamic cloud storage: A colonel blotto game approach. *IEEE Internet of Things Journal,* 5(6), 4250–4261. https://doi.org/10.1109/jiot.2018.2844878

MITRE. (2020, March 24). *Input capture*. MITRE. https://attack.mitre.org/techniques/T1056/003/

MITRE. (2022, April 18). *Boot or logon autostart execution*. MITRE. https://attack.mitre.org/techniques/T1547/

MITRE. (2023a). *Caldera*. MITRE. https://caldera.mitre.org/

MITRE. (2023b). *MITRE ATT&CK v13*. https://attack.mitre.org/resources/updates/updates-april-2023/

Mughal, A. A. (2018). The art of cybersecurity: Defense in depth strategy for robust protection. *International Journal of Intelligent Automation and Computing, 1*(1), 1–20.

Murugiah, S., & Scarfone, K. (2016). *NIST SP 800-154 Guide to data-centric system threat modeling*. National Institute of Standards and Technology.

Myneni, S., Chowdhary, A., Sabur, A., Sengupta, S., Agrawal, G., Huang, D., & Kang, M. (2020). DAPT 2020 - Constructing a benchmark dataset for advanced persistent threats. *Deployable Machine Learning for Security Defense,* 138–163. https://doi.org/10.1007/978-3-030-59621-7_8

Nakashima, E., & Timberg, C. (2017). NSA officials worried about the day its potent hacking tool would get loose. Then it did. *Washington Post.* https://www.washingtonpost.com/business/technology/nsa-officials-worried-about-the-day-its-potent-hacking-tool-would-get-loose-then-it-did/2017/05/16/50670b16-3978-11e7-a058-ddbb23c75d82_story.html

National Science and Technology Council. (2020). *Artificial intelligence and cybersecurity: opportunities and challenges technical workshop summary report*. Networking & Information Technology Research and Development Subcommittee of the National Science and Technology Council. https://www.nitrd.gov/pubs/ai-cs-tech-summary-2020.pdf

Nawrocki, M., Wahlisch, M., Scmidt, T., Keil, C., Schonfelder, J. (2016, August). *A survey on honeypot software and data analysis*. Free University Berlin.

Nmap Project. (n.d.). *Nmap reference guide* (7.92 ed.). https://nmap.org/book/

Noureddine, Mohammad A., et al. "A game-theoretic approach to respond to attacker lateral movement." *Decision and Game Theory for Security: 7th International Conference, GameSec 2016, USA, Proceedings* 7. Springer International Publishing, 2016.

Nweke, L., & Wolthusen, S. (2020). A review of asset-centric threat modelling approaches. *International Journal of Advanced Computer Science and Applications,* 11(2), *1-7.* Office of the Director of National Intelligence https://www.dni.gov/index.php/gt2040-home

Oliveira, A., Shuper, A., Frimark, I., Logan, M. Kinger, P., Manral, V., Weizman, Y. (2022, April 1). *Deploy container*. MITRE. https://attack.mitre.org/techniques/T1610/

Osborne, M. (2004). *An introduction to game theory* (Vol. 3, No. 3). Oxford University Press.

Packet Tracer. (2022). *Packet Tracer version 8.2.0.0162*. Cisco.

Panjwani, S., Tan, S., Jarrin, K., & Cukier, M. (2005). An experimental evaluation to determine if port scans are precursors to an attack. *In 2005 International Conference on Dependable Systems and Networks* (pp. 602–611). IEEE.

Park, J. (2021). The Lazarus group: The cybercrime syndicate financing the North Korean state. *Harvard International Review, 42*(2), 34-39.

Park, K., Ahn, B., Kim, J., Won, D., Noh, Y., Choi, J., & Kim, T. (2021). An advanced persistent threat-style cyberattack testbed for distributed energy resources. *IEEE Xplore.* https://doi.org/10.1109/DMC51747.2021.9529953

Parunak, H., Greanya, J., McCarthy, P., Morell, J. A., Nadella, S., & Sappelsa, L. (2021). SCAMP's stigmergic model of social conflict. *Computational and Mathematical Organization Theory.* https://doi.org/10.1007/s10588-021-09347-8

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85), 2825-2830. Retrieved from http://jmlr.org/papers/v12/pedregosa11a.html

Pingios, A., Lee, B., Naeem, D. (2022). *Magic Hound*. MITRE Groups. https://attack.mitre.org/groups/G0059/

Pipyros, K., Mitrou, L., Gritzalis, D., & Apostolopoulos, T. (2014, July). A cyber attack evaluation methodology. *In Proc. of the 13th European Conference on Cyber Warfare and Security* (pp. 264–270).

Quintero-Bonilla, S., & Martín del Rey, A. (2020). A new proposal on the advanced persistent threat: A survey. *Applied Sciences, 10*(11), *3874.* https://doi.org/10.3390/app10113874

Rass, S., König, S., & Schauer, S. (2017). Defending against advanced persistent threats using game-theory. *PLOS ONE, 12*(1), *e0168675.* https://doi.org/10.1371/journal.pone.0168675

Rescorla, E., & Korver, B. (2003). *Guidelines for writing RFC text on security considerations*. https://doi.org/10.17487/rfc3552

Roberson, B. (2006). The colonel blotto game. *Economic Theory, 29*(1), 1–24. https://doi.org/10.1007/s00199-005-0071-5

Rowe, N. (2004a). A model of deception during cyber-attacks on information systems. *IEEE First Symposium on Multi-Agent Security and Survivability, USA* (pp. 21–30). doi: 10.1109/MASSUR.2004.1368414

Rowe, N. (2004b). *Designing good deceptions in defense of information systems*. Computer Security Applications Conference, USA.

Rowe, N. (2007a). *Planning cost-effective deceptive resource denial in defense to cyber-attacks*. http://hdl.handle.net/10945/36579

Rowe, N. (2007b). Finding logically consistent resource-deception plans for defense in cyberspace. *21st International Conference on Advanced Information Networking and Applications Workshops.* https://doi.org/10.1109/ainaw.2007.186

Rowe, N., & Rrushi, J. (2016). *Introduction to cyber deception.* doi:10.1007/978-3-319-41187-3

Roy, S., Sharmin, N., Acosta, J., Kiekintveld, C., & Laszka, A. (2022). Survey and taxonomy of adversarial reconnaissance techniques. *ACM Computing Surveys, 55*(6), *1–38.* https://doi.org/10.1145/3538704

Rubio, J., Alcaraz, C., & Lopez, J. (2020). Game theory-based approach for defense against APTs. *Applied Cryptography and Network Security*, 297–320. https://doi.org/10.1007/978-3-030-57878-7_15

Schmitt, M. (2017). *Tallinn manual 2.0 on the international law applicable to cyber operations*. Cambridge University Press.

Shen, F., Perigo, L., & Curry, J. (2023). SR2APT: A detection and strategic alert response model against multistage APT attacks. *Security and Communication Networks, 2023.*

Shenwen, L., Yingbo, L., & Xiongjie, D. (2015). Study and research of APT detection technology based on big data processing architecture. 2015 *IEEE 5th International Conference on Electronics Information and Emergency Communication, China* (pp. 313–316). doi: 10.1109/ICEIEC.2015.7284547.

Siddiqui, S., Khan, M., Ferens, K., & Kinsner, W. (2016). Detecting advanced persistent threats using fractal dimension-based machine learning classification. *Proceedings of the 2016 ACM on International Workshop on Security and Privacy Analytics.* https://doi.org/10.1145/2875475.2875484

Soesanto, S. (2021). *The 19th of July: Divided or united in cyberspace? From the EU and NATO to Five Eyes and Japan*. Royal Institute of Spain.

Sonchack, J., Dubey, A., Aviv, A., Smith, J., & Keller, E. (2016). Timing-based reconnaissance and defense in software-defined networks. *Annual Computer Security Applications Conference.* https://doi.org/10.1145/2991079.2991081

Sternstein, J., Wee, M., Somasamudram, P., Sarukkai, S., Farooqh, S., Weizman,Y. (2022). *Valid accounts*. MITRE. https://attack.mitre.org/techniques/T1078/

Strom, B., Microsoft 365 Defender, Geesaman, B., Williams. (2022, October 21). *Indicator removal*. MITRE. https://attack.mitre.org/techniques/T1070/

Suryateja, P. S. (2018). Threats and vulnerabilities of cloud computing: A review. *International Journal of Computer Sciences and Engineering, 6*(3), 297–302. https://doi.org/10.26438/ijcse/v6i3.297302

Tatam, M., Shanmugam, B., Azam, S., & Kannoorpatti, K. (2021). A review of threat modelling approaches for APT-style attacks. *Heliyon, 7*(1), e05969. https://doi.org/10.1016/j.heliyon.2021.e05969

Tian, W., Ji, X., Liu, W., Liu, G., Zhai, J., Dai, Y., & Huang, S. (2020). Prospect theoretic study of honeypot defense against advanced persistent threats in power grid. *IEEE Access, 8*, *64075-64085.*

Tian, W., Ji, X., Liu, W., Zhai, J., Liu, G., Dai, Y., & Huang, S. (2019). Honeypot game-theoretical model for defending against APT attacks with limited resources in cyber-physical systems. *ETRI Journal, 41*(5), 585–598. https://doi.org/10.4218/etrij.2019-0152

Tsai, W. T., Bai, X. Y., & Huang, Y. (2014). Software-as-a-service (SAAS): Perspectives and challenges. *Science China Information Sciences, 57*(5), 1–15. https://doi.org/10.1007/s11432-013-5050-z

Ullah, F., Edwards, M., Ramdhany, R., Chitchyan, R., Babar, M., & Rashid, A. (2018). Data exfiltration: A review of external attack vectors and countermeasures. *Journal of Network and Computer Applications, 101*, 18–54. https://doi.org/10.1016/j.jnca.2017.10.016

U.S. Government. (1986). Computer Fraud and Abuse Act, 18 U.S.C. § 1030. https://www.law.cornell.edu/uscode/text/18/1030

U.S. White House. (2023). *National cybersecurity strategy*. https://www.whitehouse.gov/wp-content/uploads/2023/03/National-Cybersecurity-Strategy-2023.pdf

U.S. White House. (2022). *National security strategy*. https://www.whitehouse.gov/wp-content/uploads/2022/10/Biden-Harris-Administrations-National-Security-Strategy-10.2022.pdf

Van Dijk, M., Juels, A., Oprea, A., & Rivest, R. (2013). FlipIt: The game of "stealthy takeover". *Journal of Cryptology, 26*, 655–713.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(11).

Virvilis, N., Vanautgaerden, B., & Serrano, O. (2014). Changing the game: The art of deceiving sophisticated attackers. *2014 6th International Conference on Cyber Conflict (CyCon 2014), Estonia* (pp. 87–97). doi:10.1109/CYCON.2014.6916397.

Wang, W., Bickford, J., Murynets, I., Subbaraman, R., G. Forte, A., & Singaraju, G. (2013). Detecting targeted attacks by multilayer deception. *Journal of Cyber Security and Mobility, 2*(2), 175–199. https://doi.org/10.13052/jcsm2245-1439.224

Wang, C., & Lu, Z. (2018). Cyber deception: Overview and the road ahead. *IEEE Security & Privacy,* 16(2), *80–85.* https://doi.org/10.1109/msp.2018.1870866

Wang, J., Herath, T., Chen, R., Vishwanath, A., & Rao, H. (2012). Research article phishing susceptibility: An investigation into the processing of a targeted spear phishing email. *IEEE Transactions on Professional Communication, 55*(4), 345–362. https://doi.org/10.1109/tpc.2012.2208392

Wang, L., & Wu, D. (2016). Moving target defense against network reconnaissance with software defined networking. *Lecture Notes in Computer Science* (pp. 203–217). Springer Science and Business Media. https://doi.org/10.1007/978-3-319-45871-7_13

Wang, X., Zheng, K., Niu, X., Wu, B., & Wu, C. (2016). Detection of command and control in advanced persistent threat based on independent access. *2016 IEEE International Conference on Communications (ICC), Malaysia* (pp. 1–6). doi: 10.1109/ICC.2016.7511197.

Winther, P. (2021, October 18). *Phishing: Spearphishing attachment*. MITRE. https://attack.mitre.org/techniques/T1566/001/

Winther, P., Simmon, R., & Salla, S. (2022, March 8). *Phishing for information*. MITRE. https://attack.mitre.org/techniques/T1598/

Xiao, Z., & Xiao, Y. (2013). Security and privacy in cloud computing. *IEEE Communications Surveys & Tutorials, 15*(2), 843–859. https://doi.org/10.1109/surv.2012.060912.00182

Xi, B., & Kamhoua, C. (2020). A hypergame-based defense strategy toward cyber deception in internet of battlefield things. *In Modeling and Design of Secure Internet of Things* (pp. 59–77). IEEE. doi: 10.1002/9781119593386.ch3.

Xu, J., Guo, P., Zhao, M., Erbacher, R., Zhu, M., & Liu, P. (2014). Comparing different moving target defense techniques. *Proceedings of the First ACM Workshop on Moving Target Defense.* https://doi.org/10.1145/2663474.2663486

Yang, L., Li, P., Yang, X., & Tang, Y. (2018). A risk management approach to defending against the advanced persistent threat. *IEEE Transactions on Dependable and Secure Computing, 1–1.* https://doi.org/10.1109/tdsc.2018.2858786

Yen, S., & Lee, Y. (2006). Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset. *Intelligent Control and Automation: International Conference on Intelligent Computing,* 731–740. Springer Berlin Heidelberg.

Zhang, L., & Thing, V. (2021). Three decades of deception techniques in active cyber defense - Retrospect and outlook. *Computers & Security,* 106, *102288.* https://doi.org/10.1016/j.cose.2021.102288

Zhang, M., Zheng, Z., & Shroff, N. (2020). Defending against stealthy attacks on multiple nodes with limited resources: A game-theoretic analysis. *IEEE Transactions on Control of Network Systems, 7*(4), 1665–1677. https://doi.org/10.1109/tcns.2020.2993281

Zhao, M., An, B., & Kiekintveld, C. (2016). Optimizing personalized email filtering thresholds to mitigate sequential spear phishing attacks. *Proceedings of the AAAI Conference on Artificial Intelligence, 30*(1). https://doi.org/10.1609/aaai.v30i1.10030

Zhu, M., Anwar, A., Wan, Z., Cho, J., Kamhoua, C., & Singh, M. (2021). A survey of defensive deception: Approaches using game theory and machine learning. *IEEE Communications Surveys & Tutorials, 23*(4)*,* 2460–2493. https://doi.org/10.1109/comst.2021.3102874

Zhu, Q., & Rass, S. (2018). On multi-phase and multi-stage game-theoretic modeling of advanced persistent threats. *IEEE Access, 6,* 13958–13971. https://doi.org/10.1109/access.2018.2814481

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California